## Declarations

**Ports:**

```
ALGO                    : std_logic_vector(191 DOWNTO 0)
ALGO_S                  : std_logic_vector(5 DOWNTO 0)
BCRES                   : std_logic
BX                      : std_logic_vector(11 DOWNTO 0)
CLK80_TO_FDL            : std_logic
CLK_FBIN_FDL            : std_logic
CLK_TO_FDL              : std_logic
EN_FDL                  : std_logic
INACTIVE                : std_logic
L1A                     : std_logic
L1RESET                 : std_logic
RDRQST                  : std_logic
RESET_FDL_CHIP          : std_logic
STAT_11                 : std_logic_vector(3 DOWNTO 0)
STAT_12                 : std_logic_vector(3 DOWNTO 0)
STAT_13                 : std_logic_vector(3 DOWNTO 0)
STAT_14                 : std_logic_vector(3 DOWNTO 0)
STAT_15                 : std_logic_vector(3 DOWNTO 0)
STAT_17                 : std_logic_vector(3 DOWNTO 0)
STAT_18                 : std_logic_vector(3 DOWNTO 0)
STAT_19                 : std_logic_vector(3 DOWNTO 0)
STAT_20                 : std_logic_vector(3 DOWNTO 0)
STAT_21                 : std_logic_vector(3 DOWNTO 0)
STAT_9                  : std_logic_vector(3 DOWNTO 0)
STROB                   : std_logic_vector(2 DOWNTO 0)
TTRIG                   : std_logic_vector(31 DOWNTO 0)
VA_I                    : std_logic_vector(19 DOWNTO 1)
WR_FDL                  : std_logic
A_BEAM                  : std_logic_vector(1 DOWNTO 0)
CLK_DAQ                 : std_logic
CLK_EVM                 : std_logic
CLK_FBOUT_FDL           : std_logic
CLK_LOCKED_FDL          : std_logic
DAQ_D                   : std_logic_vector(27 DOWNTO 0)
ERR_PAN                 : std_logic
EVM_D                   : std_logic_vector(27 DOWNTO 0)
FINOR_PAN               : std_logic
FIN_OR                  : std_logic_vector(15 DOWNTO 0)
GCLK0S                  : std_ulogic
GT_STATUS               : std_logic_vector(3 DOWNTO 0)
IRQ_X                   : std_logic
NBERR_FDL               : std_logic
NDTACK_FDL              : std_logic
NEN_BACKRECV            : std_logic
NEN_CHAN_D              : std_logic
NEN_CHAN_E              : std_logic
TECH_TRIG               : std_logic_vector(15 DOWNTO 0)
TTRIG_PAN               : std_logic
dummy_out               : STD_ULOGIC
FDLTCS                  : std_logic_vector(3 DOWNTO 0)
FDLU_TCS                : std_logic_vector(3 DOWNTO 0)
TIMFDL                  : std_logic_vector(15 DOWNTO 0)
VD_I                    : std_logic_vector(31 DOWNTO 0)
```

**Diagram Signals:**

```
SIGNAL BCRES_i                : std_logic
SIGNAL BERR                   : std_logic
SIGNAL CLK0                   : std_ulogic := '0'
SIGNAL CLK180                 : std_ulogic := '0'
SIGNAL CLK2X                  : std_ulogic := '0'
SIGNAL CLK2X180               : std_ulogic := '0'
SIGNAL CLKIN                  : std ulogic := '0'
```

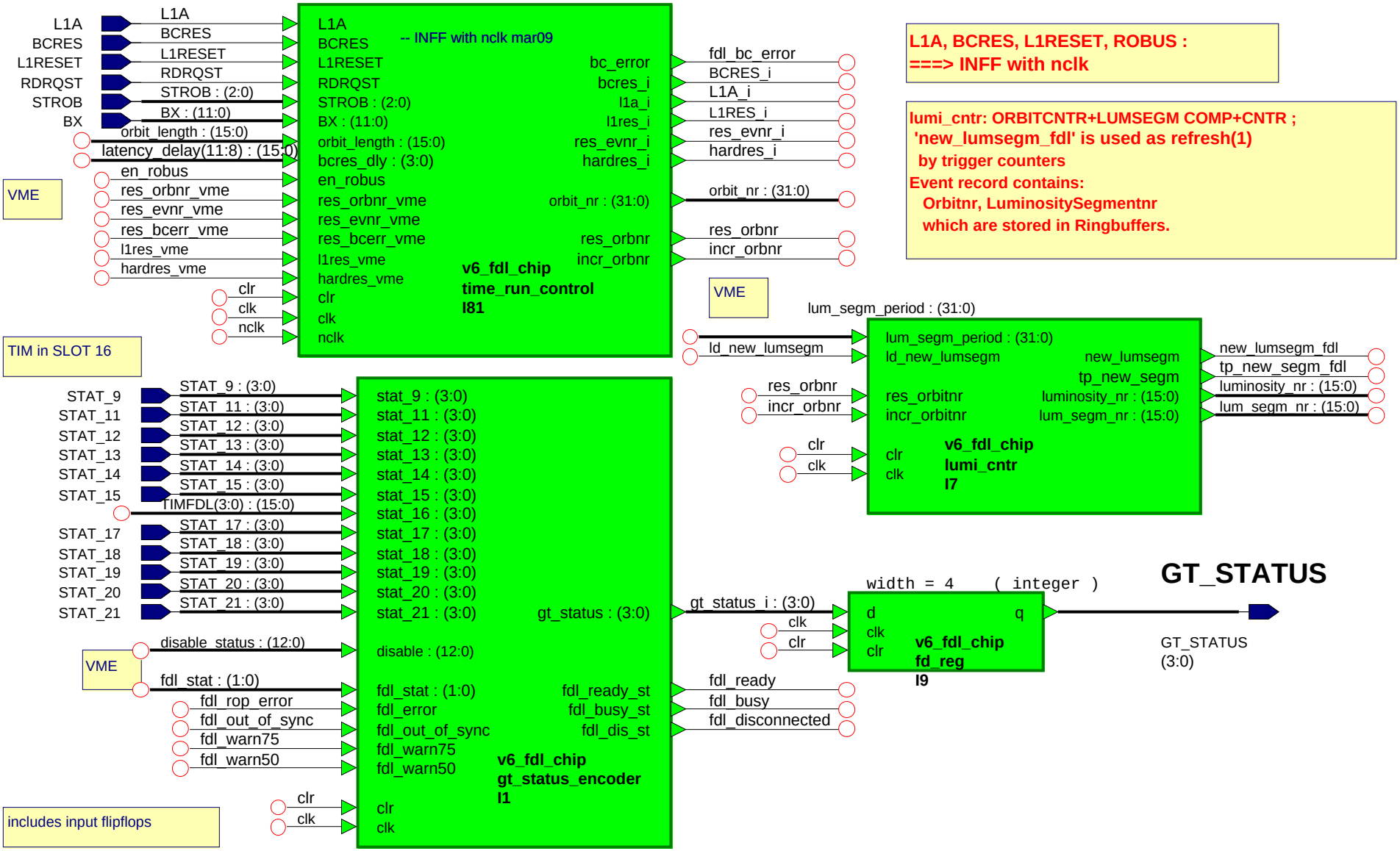
**Package List**

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
USE IEEE.VITAL_Timing.all;
LIBRARY unisim;
USE unisim.VCOMPONENTS.all;
USE IEEE.VITAL_Primitives.all;

LIBRARY fdl_types;
USE fdl_types.fdl.all;
LIBRARY STD;
USE STD.TEXTIO.all;
USE unisim.VPKG.all;
```

**Generics are defined in Symbol:**
**(click 'interface' slide)**
**no_of_algo = 128**
**no_of_ttrig = 64**

| <company name> | | Project: | fdl_chip |
|---|---|---|---|
| Title: | <enter diagram title here> | <enter comments here> | |
| Path: | v6_fdl_chip/fdl_chip/struct | | |
| Edited: | by taurok on 29 Sep 2009 | | |

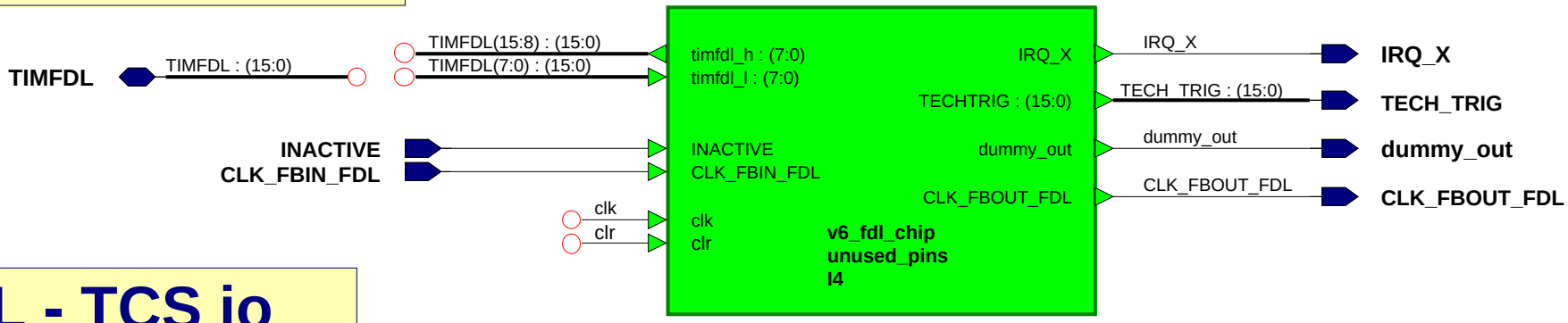

L1A
BCRES
L1RESET
RDRQST
STROB : (2:0)
BX : (11:0)
orbit_length : (15:0)
latency_delay(11:8) : (15:0)
en_robus
res_orbnr_vme
res_evnr_vme
res_bcerr_vme
l1res_vme
hardres_vme
clr
clk
nclk
-- INFF with nclk mar09
v6_fdl_chip
time_run_control
I81
bc_error
bcres_i
l1a_i
l1res_i
res_evnr_i
hardres_i
orbit_nr : (31:0)
res_orbnr
incr_orbnr
fdl_bc_error
BCRES_i
L1A_i
L1RES_i
res_evnr_i
hardres_i
VME
TIM in SLOT 16
L1A, BCRES, L1RESET, ROBUS :
===> INFF with nclk
lumi_cntr: ORBITCNTR+LUMSEGM COMP+CNTR ;
'new_lumsegm_fdl' is used as refresh(1)
by trigger counters
Event record contains:
Orbitnr, LuminositySegmentnr
which are stored in Ringbuffers.
lum_segm_period : (31:0)
ld_new_lumsegm
res_orbitnr
incr_orbitnr
new_lumsegm
tp_new_segm
luminosity_nr : (15:0)
lum_segm_nr : (15:0)
v6_fdl_chip
lumi_cntr
I7
new_lumsegm_fdl
tp_new_segm_fdl
luminosity_nr : (15:0)
lum_segm_nr : (15:0)
STAT_9 : (3:0)
STAT_11 : (3:0)
STAT_12 : (3:0)
STAT_13 : (3:0)
STAT_14 : (3:0)
STAT_15 : (3:0)
TIMFDL(3:0) : (15:0)
STAT_17 : (3:0)
STAT_18 : (3:0)
STAT_19 : (3:0)
STAT_20 : (3:0)
STAT_21 : (3:0)
disable_status : (12:0)
fdl_stat : (1:0)
fdl_rop_error
fdl_out_of_sync
fdl_warn75
fdl_warn50
disable : (12:0)
gt_status : (3:0)
fdl_error
fdl_ready_st
fdl_busy_st
fdl_dis_st
v6_fdl_chip
gt_status_encoder
I1
includes input flipflops
gt_status_i : (3:0)
width = 4 ( integer )
d
q
v6_fdl_chip
fd_reg
I9
GT_STATUS
GT_STATUS
(3:0)
fdl_ready
fdl_busy
fdl_disconnected

# FDL - TIM io

**TIMFDL(7:0) ---> input from TIM**
**7 – 4: dummy L1_RES;**
**3 - 0: TIM_FASTSIG(3:0)**
**TIMFDL(15:8) <--- output to TIM ="X00"**

# UNUSED PINS

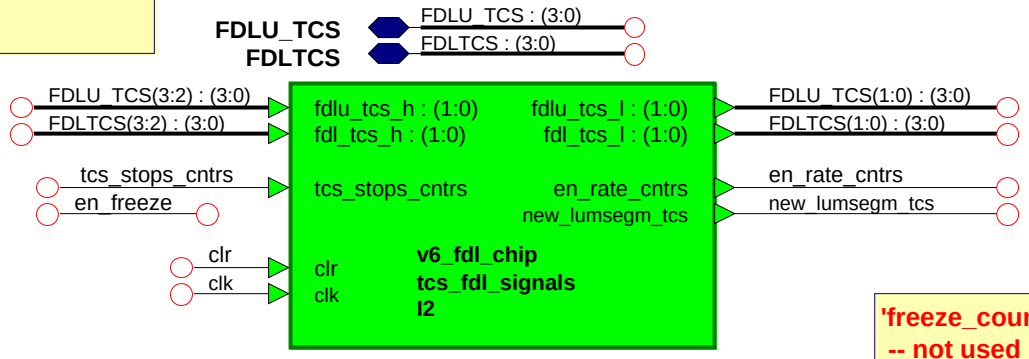
**CLK80_TO_FDL** — **not connected**




# FDL - TCS io

```
-- FDLUTCS (1:0) <--- output to TCS ="00"
-- FDLUTCS (3:2) ---> input from TCS
--         Bit 3: 'inh_fdl_cntrs'
--         Bit 2: 'new_lum_segm' => refresh_from_tcs

-- FDLTCS  (1:0) <--- output to TCS ="01" diff. signal pair
-- FDLTCS  (3:2) ---> input from TCS
--         Bit 3: '0'; -- free differential +signal
--         Bit 2: '1'; -- free differential -signal
```



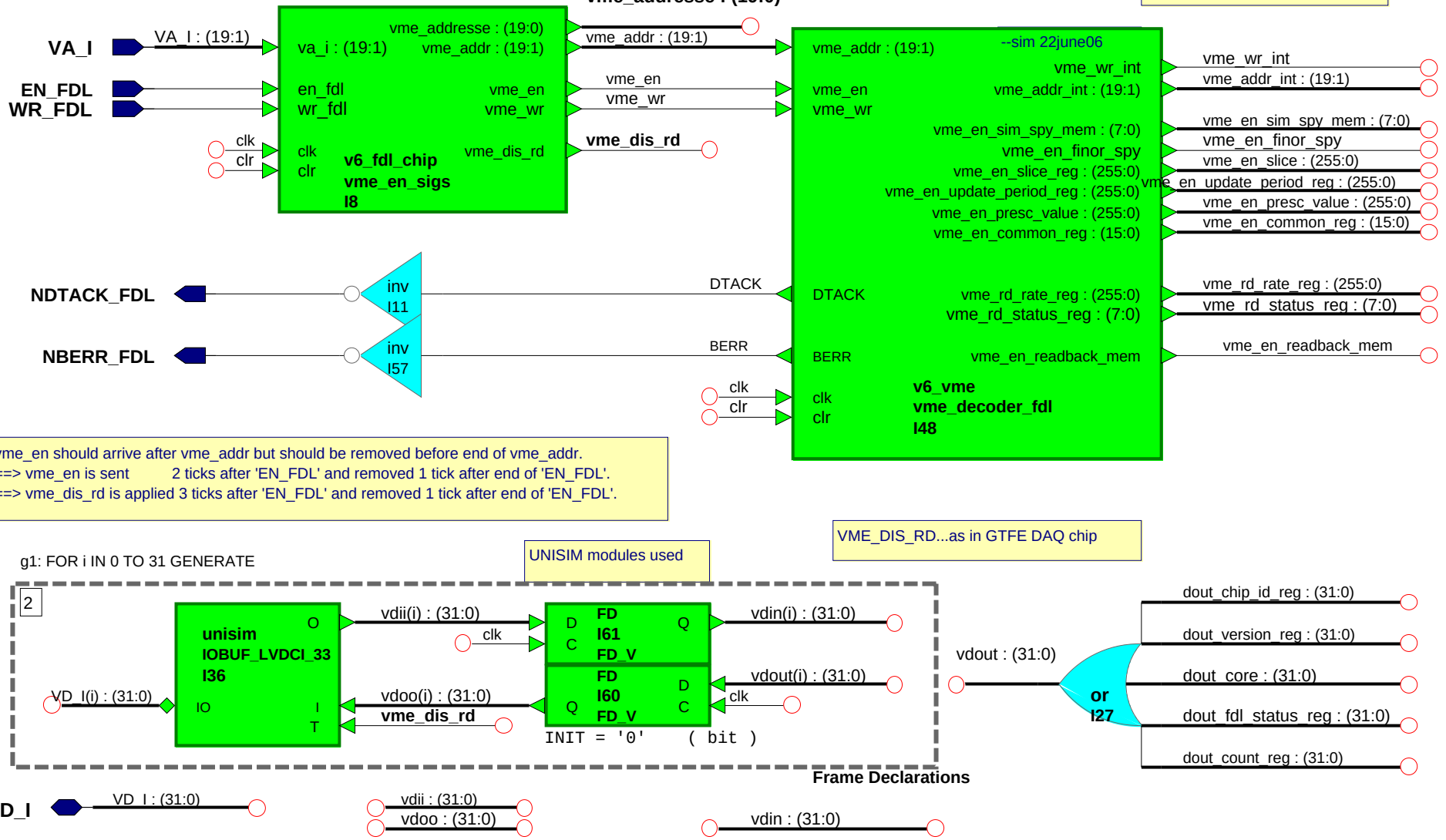

**en_freeze ==>'freeze_counters'**
**... not implemented fdl_core**

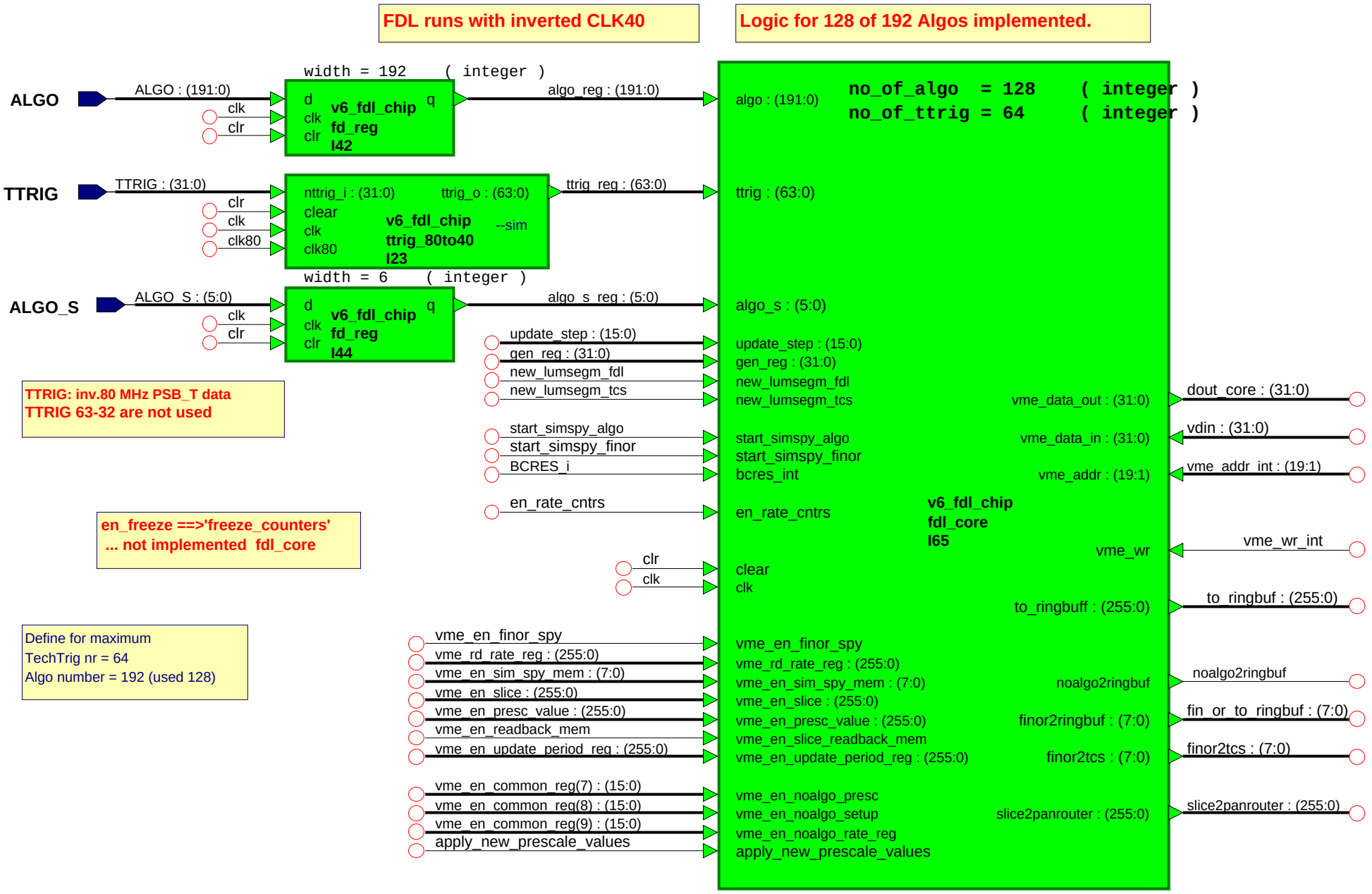
**--> to 'fdl_core'**

**'freeze_counters' ... not implemented**
**-- not used in fdl_core**

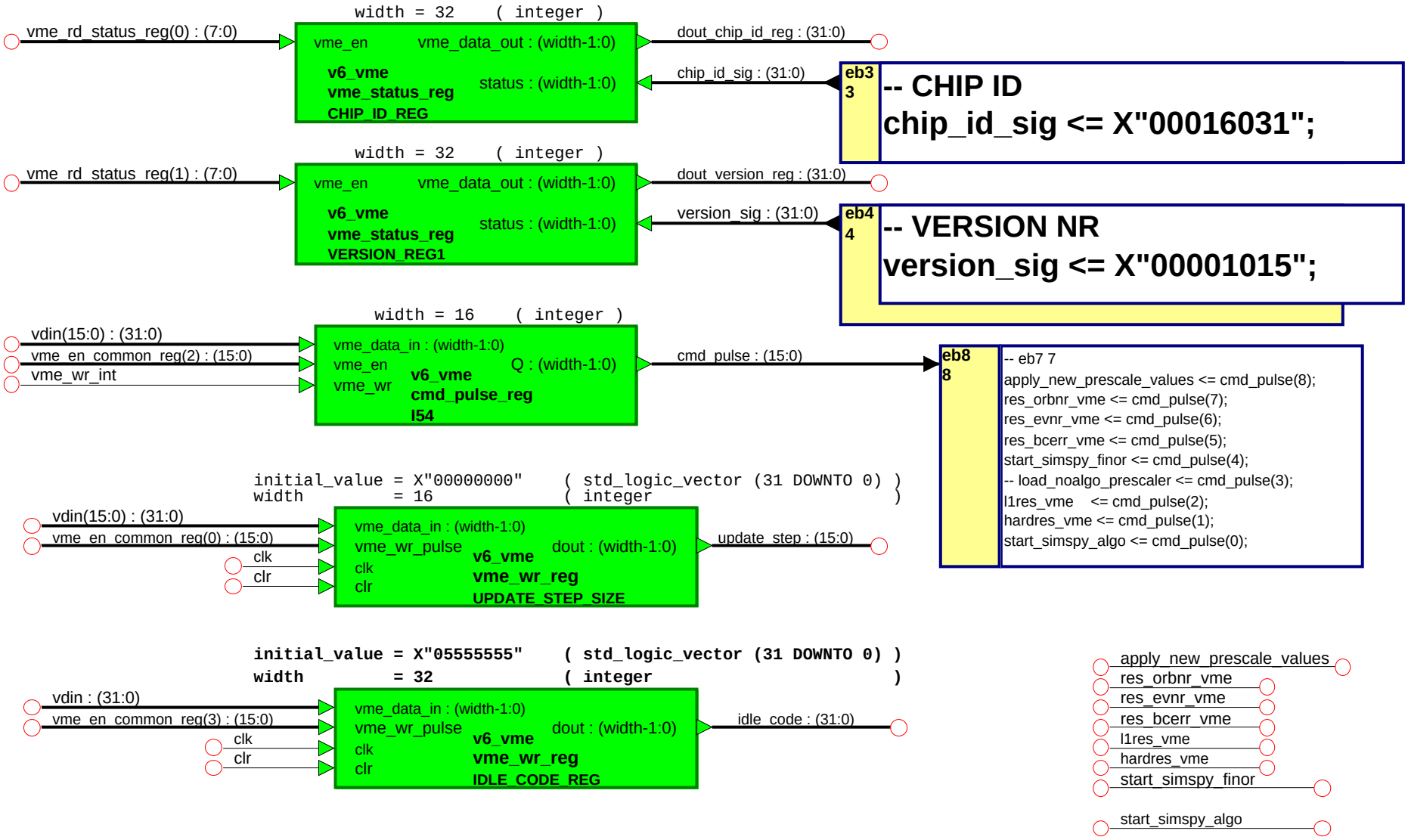
```
SIGNAL DTACK                    : std_logic
SIGNAL L1A_i                    : std_logic
SIGNAL L1RES_i                  : std_logic
SIGNAL algo_reg                 : std_logic_vector(191 DOWNTO 0)
SIGNAL algo_s_reg               : std_logic_vector(5 DOWNTO 0)
SIGNAL apply_new_prescale_values : std_logic
SIGNAL bitvec2pan               : std_logic_vector(255 DOWNTO 0)
SIGNAL board_id                 : std_logic_vector(15 DOWNTO 0)
SIGNAL chan_link_rec            : std_logic_vector(27 DOWNTO 0)
SIGNAL chip_id_sig              : std_logic_vector(31 DOWNTO 0)
SIGNAL clk                      : std_logic
SIGNAL clk80                    : std_logic
SIGNAL clr                      : std_logic
SIGNAL cmd_pulse                : std_logic_vector(15 DOWNTO 0)
SIGNAL din0                     : std_logic
SIGNAL disable_status           : std_logic_vector(12 DOWNTO 0)
SIGNAL dout_chip_id_reg         : std_logic_vector(31 DOWNTO 0)
SIGNAL dout_core                : std_logic_vector(31 DOWNTO 0)
SIGNAL dout_count_reg           : std_logic_vector(31 DOWNTO 0)
SIGNAL dout_fdl_status_reg      : std_logic_vector(31 DOWNTO 0)
SIGNAL dout_version_reg         : std_logic_vector(31 DOWNTO 0)
SIGNAL en_freeze                : std_logic
SIGNAL en_rate_cntrs            : std_logic
SIGNAL en_robus                 : std_logic
SIGNAL en_slice4pan             : std_logic_vector(255 DOWNTO 0)
SIGNAL err2pan                  : STD_ULOGIC
SIGNAL fdl_bc_error             : std_logic
SIGNAL fdl_busy                 : std_logic
SIGNAL fdl_disconnected         : std_logic
SIGNAL fdl_out_of_sync          : std_logic
SIGNAL fdl_ready                : std_logic
SIGNAL fdl_rop_error            : std_logic
SIGNAL fdl_stat                 : std_logic_vector(1 DOWNTO 0)
SIGNAL fdl_warn50               : std_logic
SIGNAL fdl_warn75               : std_logic
SIGNAL fin_or_to_ringbuf        : std_logic_vector(7 DOWNTO 0)
SIGNAL finor2outbuf             : std_logic_vector(7 DOWNTO 0)
SIGNAL finor2pan                : std_logic
SIGNAL finor2tcs                : std_logic_vector(7 DOWNTO 0)
SIGNAL finor_ff                 : std_logic_vector(7 DOWNTO 0)
SIGNAL gen_reg                  : std_logic_vector(31 DOWNTO 0)
SIGNAL gnd_dcm                  : std_logic
SIGNAL gsr                      : std_ulogic := '0'
SIGNAL gt_status_i              : std_logic_vector(3 DOWNTO 0)
SIGNAL hardres_i                : std_logic
SIGNAL hardres_vme              : std_logic
SIGNAL idle_code                : std_logic_vector(31 DOWNTO 0)
SIGNAL incr_orbnr               : std_logic
SIGNAL l1res_vme                : std_logic
SIGNAL latency_delay            : std_logic_vector(15 DOWNTO 0)
SIGNAL ld_new_lumsegm           : std_logic
SIGNAL locked                   : std_ulogic
SIGNAL lum_segm_nr              : std_logic_vector(15 DOWNTO 0)
SIGNAL lum_segm_period          : std_logic_vector(31 DOWNTO 0)
SIGNAL luminosity_nr            : std_logic_vector(15 DOWNTO 0)
SIGNAL nclk                     : std_logic
SIGNAL nclk80                   : std_logic
SIGNAL nenbackrec               : STD_ULOGIC
SIGNAL nenchan_d                : STD_ULOGIC
SIGNAL nenchan_e                : STD_ULOGIC
SIGNAL new_lumsegm_fdl          : std_logic
SIGNAL new_lumsegm_tcs          : std_logic
SIGNAL new_prescale_version     : std_logic
SIGNAL noalgo2ringbuf           : std_logic
SIGNAL orbit_length             : std_logic_vector(15 DOWNTO 0)
```

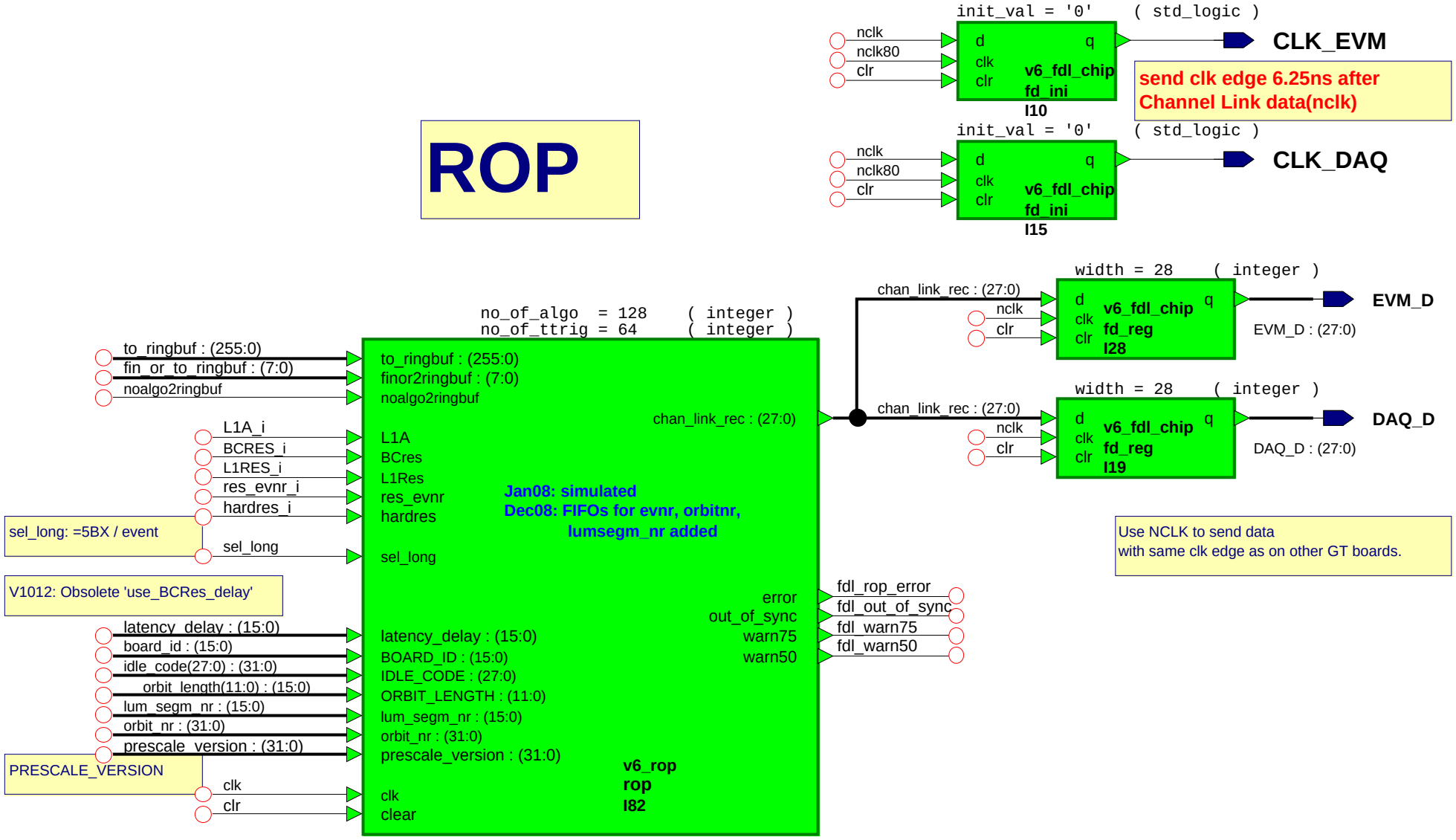
# VME Interface




FDL runs with inverted CLK40
Logic for 128 of 192 Algos implemented.
ALGO
ALGO : (191:0)
clk
clr
width = 192 ( integer )
d
clk
clr
v6_fdl_chip
fd_reg
I42
q
algo_reg : (191:0)
algo : (191:0)
no_of_algo = 128 ( integer )
no_of_ttrig = 64 ( integer )
TTRIG
TTRIG : (31:0)
clr
clk
clk80
nttrig_i : (31:0)
clear
clk
clk80
ttrig_o : (63:0)
v6_fdl_chip
ttrig_80to40
I23
--sim
ttrig_reg : (63:0)
ttrig : (63:0)
ALGO_S
ALGO_S : (5:0)
clk
clr
width = 6 ( integer )
d
clk
clr
v6_fdl_chip
fd_reg
I44
q
algo_s_reg : (5:0)
algo_s : (5:0)
update_step : (15:0)
gen_reg : (31:0)
new_lumsegm_fdl
new_lumsegm_tcs
TTRIG: inv.80 MHz PSB_T data
TTRIG 63-32 are not used
start_simspy_algo
start_simspy_finor
BCRES_i
bcres_int
vme_data_out : (31:0)
vme_data_in : (31:0)
vme_addr : (19:1)
dout_core : (31:0)
vdin : (31:0)
vme_addr_int : (19:1)
en_rate_cntrs
en_freeze ==>'freeze_counters'
... not implemented fdl_core
v6_fdl_chip
fdl_core
I65
vme_wr
vme_wr_int
clr
clk
clear
clk
to_ringbuff : (255:0)
to_ringbuf : (255:0)
Define for maximum
TechTrig nr = 64
Algo number = 192 (used 128)
vme_en_finor_spy
vme_rd_rate_reg : (255:0)
vme_en_sim_spy_mem : (7:0)
vme_en_slice : (255:0)
vme_en_presc_value : (255:0)
vme_en_readback_mem
vme_en_update_period_reg : (255:0)
noalgo2ringbuf
finor2ringbuf : (7:0)
finor2tcs : (7:0)
noalgo2ringbuf
fin_or_to_ringbuf : (7:0)
finor2tcs : (7:0)
vme_en_common_reg(7) : (15:0)
vme_en_common_reg(8) : (15:0)
vme_en_common_reg(9) : (15:0)
apply_new_prescale_values
vme_en_noalgo_presc
vme_en_noalgo_setup
vme_en_noalgo_rate_reg
slice2panrouter : (255:0)
slice2panrouter : (255:0)

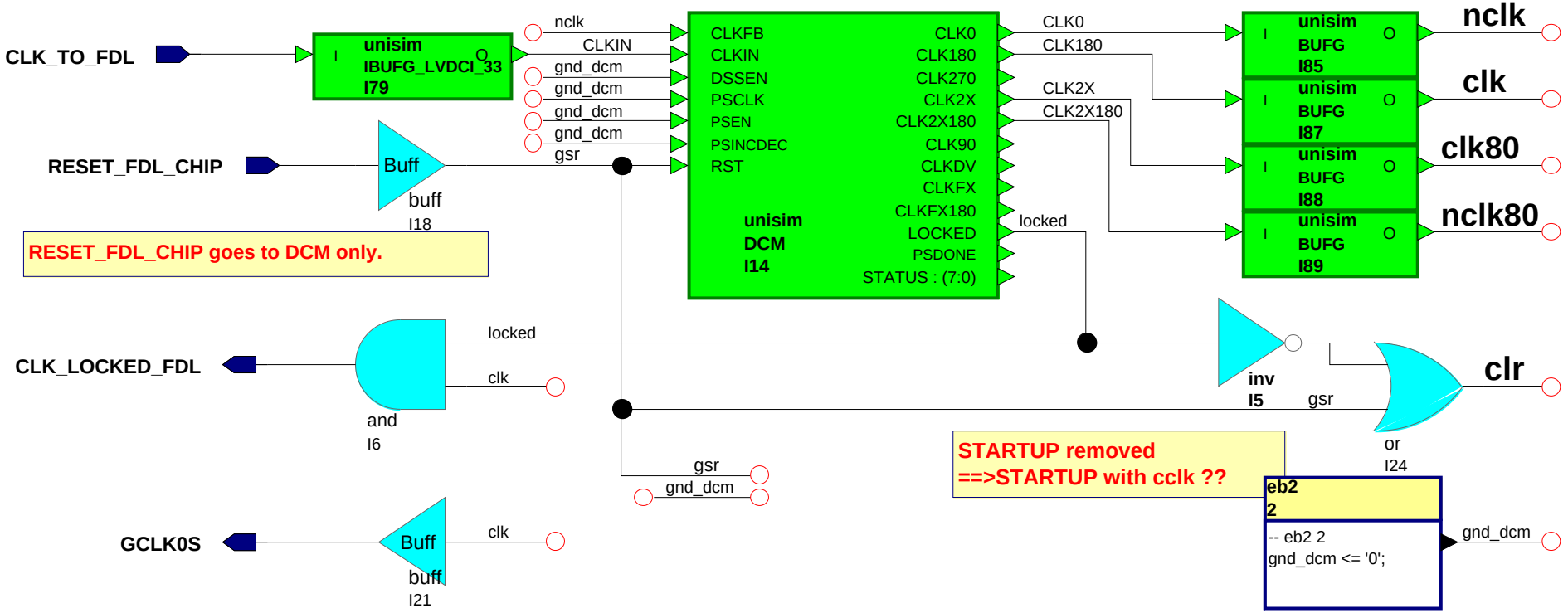
```
SIGNAL orbit_nr                : std_logic_vector(31 DOWNTO 0)
SIGNAL prescale_version        : std_logic_vector(31 DOWNTO 0)
SIGNAL q1                      : std_logic_vector(31 DOWNTO 0)
SIGNAL res_bcerr_vme           : std_logic
SIGNAL res_evnr_i              : std_logic
SIGNAL res_evnr_vme            : std_logic
SIGNAL res_orbnr               : std_logic
SIGNAL res_orbnr_vme           : std_logic
SIGNAL route_sig2pan           : std_logic_vector(31 DOWNTO 0)
SIGNAL sel_long                : std_logic
SIGNAL slice2panrouter         : std_logic_vector(255 DOWNTO 0)
SIGNAL slice_count             : std_logic_vector(15 DOWNTO 0)
SIGNAL start_simspy_algo       : std_logic
SIGNAL start_simspy_finor      : std_logic
SIGNAL status2vme              : std_logic_vector(31 DOWNTO 0)
SIGNAL tcs_stops_cntrs         : std_logic --VME
SIGNAL to_abeam                : STD_ULOGIC
SIGNAL to_ringbuf              : std_logic_vector(255 DOWNTO 0)
SIGNAL tp_new_segm_fdl         : std_logic
SIGNAL ttrig2pan               : std_logic
SIGNAL ttrig_reg               : std_logic_vector(63 DOWNTO 0)
SIGNAL update_step             : std_logic_vector(15 DOWNTO 0)
SIGNAL vcc                     : std_logic
SIGNAL vdii                    : std_logic_vector(31 DOWNTO 0)
SIGNAL vdin                    : std_logic_vector(31 DOWNTO 0)
SIGNAL vdoo                    : std_logic_vector(31 DOWNTO 0)
SIGNAL vdout                   : std_logic_vector(31 DOWNTO 0)
SIGNAL version_sig             : std_logic_vector(31 DOWNTO 0)
SIGNAL vme_addr                : std_logic_vector(19 DOWNTO 1)
SIGNAL vme_addr_int            : std_logic_vector(19 DOWNTO 1)
SIGNAL vme_addresse            : std_logic_vector(19 downto 0)
SIGNAL vme_dis_rd              : std_logic
SIGNAL vme_en                  : std_logic
SIGNAL vme_en_common_reg       : std_logic_vector(15 DOWNTO 0)
SIGNAL vme_en_finor_spy        : std_logic
SIGNAL vme_en_presc_value      : std_logic_vector(255 DOWNTO 0)
SIGNAL vme_en_readback_mem     : std_logic
SIGNAL vme_en_sim_spy_mem      : std_logic_vector(7 DOWNTO 0)
SIGNAL vme_en_slice            : std_logic_vector(255 DOWNTO 0)
SIGNAL vme_en_update_period_reg  : std_logic_vector(255 DOWNTO 0)
SIGNAL vme_rd_rate_reg         : std_logic_vector(255 DOWNTO 0)
SIGNAL vme_rd_status_reg       : std_logic_vector(7 DOWNTO 0)
SIGNAL vme_wr                  : std_logic
SIGNAL vme_wr_int              : std_logic
```

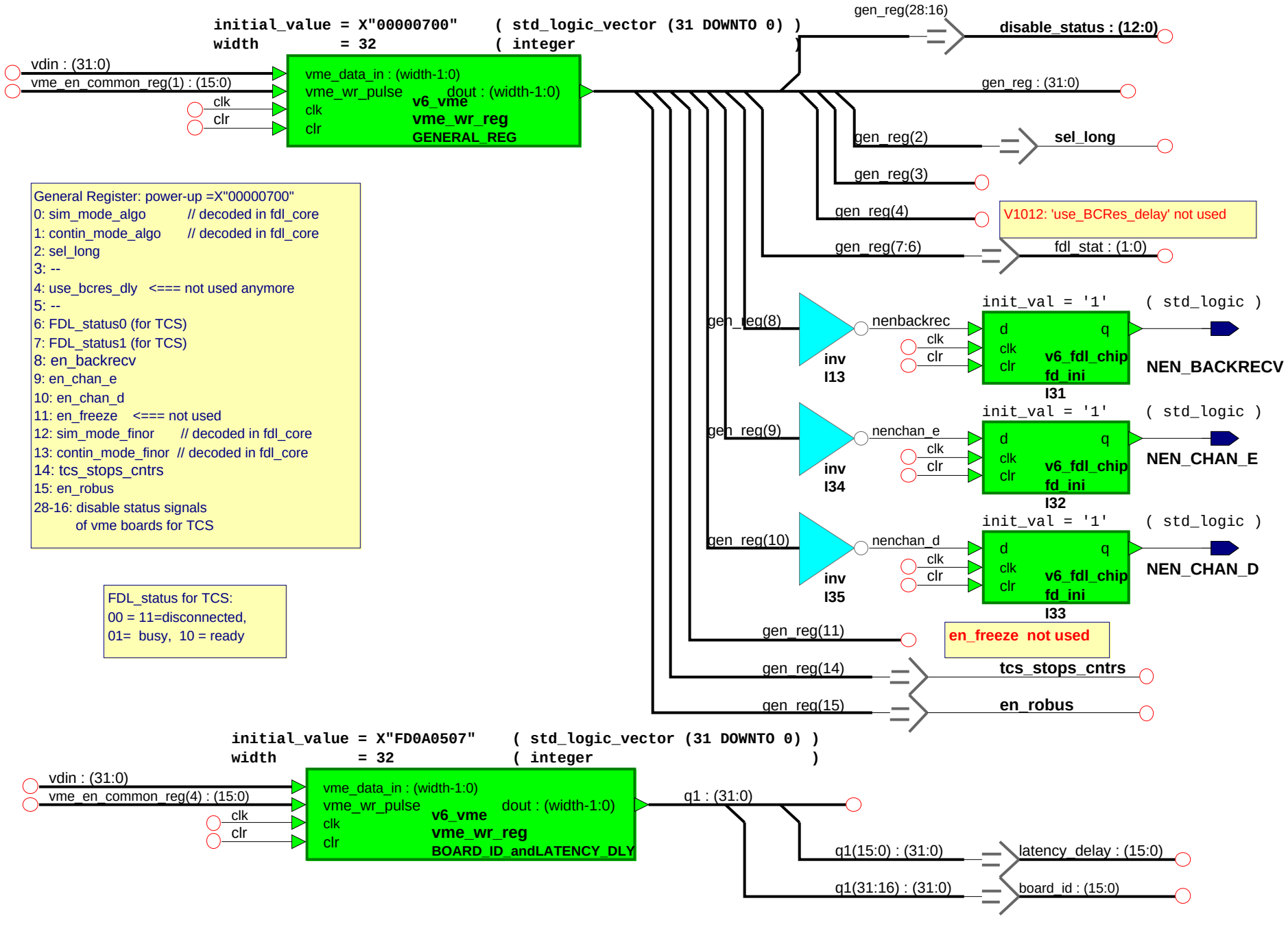

width = 32 ( integer )
vme_rd_status_reg(0) : (7:0)
vme_en
vme_data_out : (width-1:0)
v6_vme
vme_status_reg
CHIP_ID_REG
status : (width-1:0)
dout_chip_id_reg : (31:0)
chip_id_sig : (31:0)
eb33
-- CHIP ID
chip_id_sig <= X"00016031";
width = 32 ( integer )
vme_rd_status_reg(1) : (7:0)
vme_en
vme_data_out : (width-1:0)
v6_vme
vme_status_reg
VERSION_REG1
status : (width-1:0)
dout_version_reg : (31:0)
version_sig : (31:0)
eb44
-- VERSION NR
version_sig <= X"00001015";
width = 16 ( integer )
vdin(15:0) : (31:0)
vme_en_common_reg(2) : (15:0)
vme_wr_int
vme_data_in : (width-1:0)
vme_en
vme_wr
v6_vme
cmd_pulse_reg
I54
Q : (width-1:0)
cmd_pulse : (15:0)
eb88
-- eb7 7
apply_new_prescale_values <= cmd_pulse(8);
res_orbnr_vme <= cmd_pulse(7);
res_evnr_vme <= cmd_pulse(6);
res_bcerr_vme <= cmd_pulse(5);
start_simspy_finor <= cmd_pulse(4);
-- load_noalgo_prescaler <= cmd_pulse(3);
l1res_vme <= cmd_pulse(2);
hardres_vme <= cmd_pulse(1);
start_simspy_algo <= cmd_pulse(0);
initial_value = X"00000000" ( std_logic_vector (31 DOWNTO 0) )
width = 16 ( integer )
vdin(15:0) : (31:0)
vme_en_common_reg(0) : (15:0)
clk
clr
vme_data_in : (width-1:0)
vme_wr_pulse
clk
clr
v6_vme
vme_wr_reg
UPDATE_STEP_SIZE
dout : (width-1:0)
update_step : (15:0)
initial_value = X"05555555" ( std_logic_vector (31 DOWNTO 0) )
width = 32 ( integer )
vdin : (31:0)
vme_en_common_reg(3) : (15:0)
clk
clr
vme_data_in : (width-1:0)
vme_wr_pulse
clk
clr
v6_vme
vme_wr_reg
IDLE_CODE_REG
dout : (width-1:0)
idle_code : (31:0)
apply_new_prescale_values
res_orbnr_vme
res_evnr_vme
res_bcerr_vme
l1res_vme
hardres_vme
start_simspy_finor
start_simspy_algo


ROP
init_val = '0' ( std_logic )
nclk
nclk80
clr
d
q
clk
clr
v6_fdl_chip
fd_ini
I10
CLK_EVM
send clk edge 6.25ns after
Channel Link data(nclk)
I15
CLK_DAQ
width = 28 ( integer )
chan_link_rec : (27:0)
fd_reg
I28
EVM_D
EVM_D : (27:0)
I19
DAQ_D
DAQ_D : (27:0)
Use NCLK to send data
with same clk edge as on other GT boards.
no_of_algo = 128 ( integer )
no_of_ttrig = 64 ( integer )
to_ringbuf : (255:0)
fin_or_to_ringbuf : (7:0)
noalgo2ringbuf
finor2ringbuf : (7:0)
L1A_i
BCRES_i
L1RES_i
res_evnr_i
hardres_i
L1A
BCres
L1Res
res_evnr
hardres
Jan08: simulated
Dec08: FIFOs for evnr, orbitnr,
lumsegm_nr added
sel_long: =5BX / event
sel_long
V1012: Obsolete 'use_BCRes_delay'
error
out_of_sync
warn75
warn50
fdl_rop_error
fdl_out_of_sync
fdl_warn75
fdl_warn50
latency_delay : (15:0)
board_id : (15:0)
idle_code(27:0) : (31:0)
orbit_length(11:0) : (15:0)
lum_segm_nr : (15:0)
orbit_nr : (31:0)
prescale_version : (31:0)
BOARD_ID : (15:0)
IDLE_CODE : (27:0)
ORBIT_LENGTH : (11:0)
PRESCALE_VERSION
clear
v6_rop
rop
I82

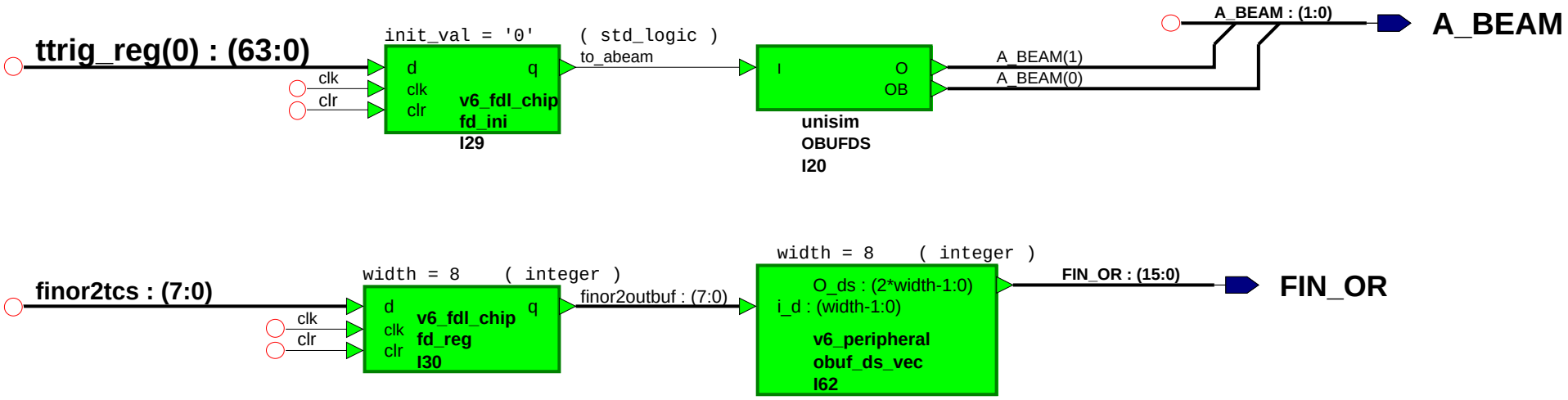
# FDL runs with inverted CLK40 to reduce GTL==>FDL==>TCS Latency.

```
CLKDV_DIVIDE           = 2.0                    ( real       )
CLKFX_DIVIDE           = 1                      ( integer    )
CLKFX_MULTIPLY         = 4                      ( integer    )
CLKIN_DIVIDE_BY_2      = false                  ( boolean    )
CLKIN_PERIOD           = 10.0                   ( real       ) --non-simulatable
CLKOUT_PHASE_SHIFT     = "NONE"                 ( string     )
CLK_FEEDBACK           = "1X"                   ( string     )
DESKEW_ADJUST          = "SYSTEM_SYNCHRONOUS"   ( string     ) --non-simulatable
DFS_FREQUENCY_MODE     = "LOW"                  ( string     )
DLL_FREQUENCY_MODE     = "LOW"                  ( string     )
DSS_MODE               = "NONE"                 ( string     ) --non-simulatable
DUTY_CYCLE_CORRECTION  = true                   ( boolean    )
FACTORY_JF             = X"C080"                ( bit_vector ) --non-simulatable
PHASE_SHIFT            = 0                      ( integer    )
SIM_MODE               = "SAFE"                 ( string     )
STARTUP_WAIT           = false                  ( boolean    ) --non-simulatable
```




initial_value = X"00000700" ( std_logic_vector (31 DOWNTO 0) )
width = 32 ( integer )
vdin : (31:0)
vme_en_common_reg(1) : (15:0)
clk
clr
vme_data_in : (width-1:0)
vme_wr_pulse
clk
clr
dout : (width-1:0)
v6_vme
vme_wr_reg
GENERAL_REG
gen_reg(28:16)
disable_status : (12:0)
gen_reg : (31:0)
gen_reg(2)
sel_long
gen_reg(3)
gen_reg(4)
V1012: 'use_BCRes_delay' not used
gen_reg(7:6)
fdl_stat : (1:0)
General Register: power-up =X"00000700"
0: sim_mode_algo // decoded in fdl_core
1: contin_mode_algo // decoded in fdl_core
2: sel_long
3: --
4: use_bcres_dly <=== not used anymore
5: --
6: FDL_status0 (for TCS)
7: FDL_status1 (for TCS)
8: en_backrecv
9: en_chan_e
10: en_chan_d
11: en_freeze <=== not used
12: sim_mode_finor // decoded in fdl_core
13: contin_mode_finor // decoded in fdl_core
14: tcs_stops_cntrs
15: en_robus
28-16: disable status signals
of vme boards for TCS
gen_reg(8)
inv
I13
nenbackrec
clk
clr
init_val = '1' ( std_logic )
d
clk
clr
q
v6_fdl_chip
fd_ini
I31
NEN_BACKRECV
gen_reg(9)
inv
I34
nenchan_e
clk
clr
init_val = '1' ( std_logic )
d
clk
clr
q
v6_fdl_chip
fd_ini
I32
NEN_CHAN_E
gen_reg(10)
inv
I35
nenchan_d
clk
clr
init_val = '1' ( std_logic )
d
clk
clr
q
v6_fdl_chip
fd_ini
I33
NEN_CHAN_D
FDL_status for TCS:
00 = 11=disconnected,
01= busy, 10 = ready
gen_reg(11)
en_freeze not used
gen_reg(14)
tcs_stops_cntrs
gen_reg(15)
en_robus
initial_value = X"FD0A0507" ( std_logic_vector (31 DOWNTO 0) )
width = 32 ( integer )
vdin : (31:0)
vme_en_common_reg(4) : (15:0)
clk
clr
vme_data_in : (width-1:0)
vme_wr_pulse
clk
clr
dout : (width-1:0)
v6_vme
vme_wr_reg
BOARD_ID_andLATENCY_DLY
q1 : (31:0)
q1(15:0) : (31:0)
latency_delay : (15:0)
q1(31:16) : (31:0)
board_id : (15:0)

# TTRIG(0) is used as A_BEAM




FINOR

V1015: FPGAdv8.2 & ISE10.1 used
ROP: New 5bx-event logic
en_rate_cntrs <-- FDLUTCS (3) ... error of V1014 corrected
vme_decoder: FF added on 'vme_rd_rate_reg', 'vme_rd_status_reg' signals
V1014: TCS sends signal to inhibit rate counters during calibration cycles.
genreg(14):=' tcs_stops_cntrs' ; fdl <--> tcs signals new definition
BCRES delays corrected to get coherent orbit+lumsegm numbers
precisely at the end of an orbit.
'freeze_counters' deleted
V1013: L1A_pulse starts at same time for 3 and 5 bc events
res_rd_addr comes 1 tick later for 5bc events to start reading from
2 locations before the L1A location.
V1012: L1A, BCRES, L1RESET, ROBUS : ===> INFF with nclk
delay_fdl_rop : latency_delays after bcres_delay
without 'use_bcres_delay' -->rop

OLD VERSIONS
V1011:
- Ringbuffer & FIFO for Event-, Orbit- and Luminosity Segment nr added
- 'rate_register_v6' replaced by 'read_rates' so that FDL and TCS deliver counter contents from the same lum_segment.
- prescale_version_nr updated concurrently with prescale factors
- 'fdl_slice' simplified, same behavior as before
- No_Algo logic updated
- Testpoint: res_orbnr
- VME logic: vme_enn with vme_en1,2,3
- Lumi Segment Length =2**20 ~93sec …0010_0000 hex

V1010:
'prescale_version' included to event record (ROP)
TTRIG: inverter moved to 40 MHz register input

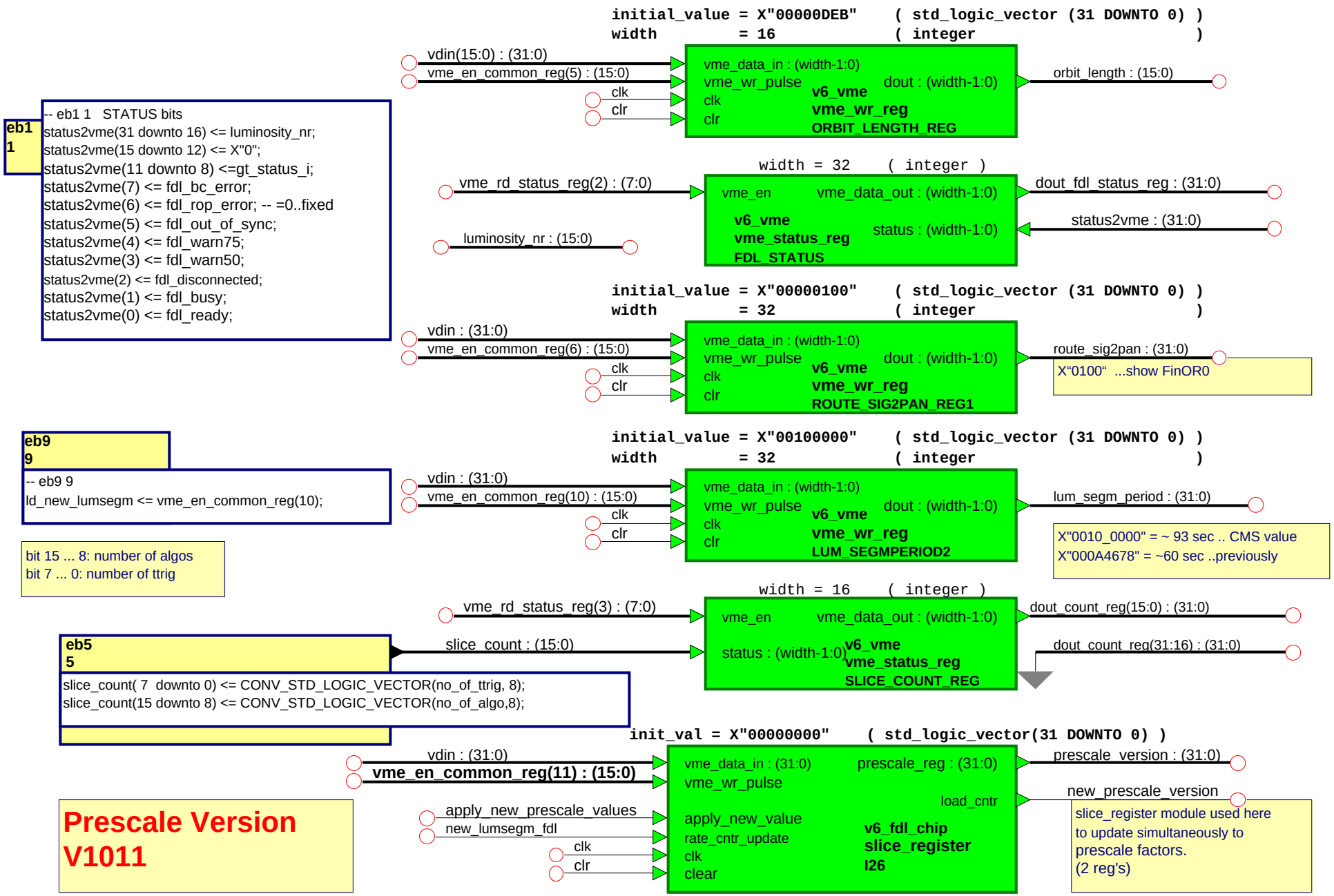
V100F:
lumi_cntr: is also reset when new lum_segment_length
has been loaded.
BCRES_i goes to testpoint

V100E:
Ringbuffer: permanent read instead of applying 'L1Apulse'

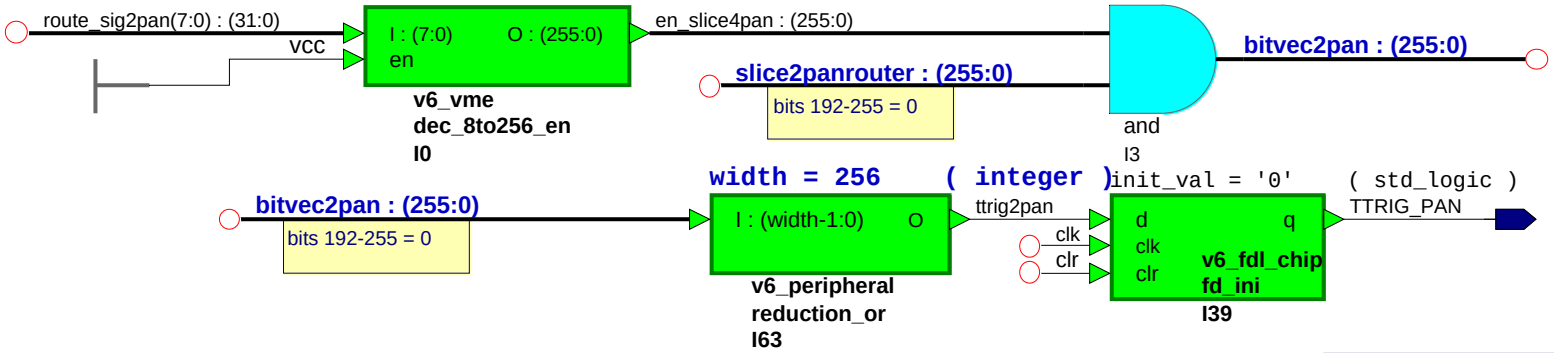
V100D:
ttrig_80to40: additional reg added

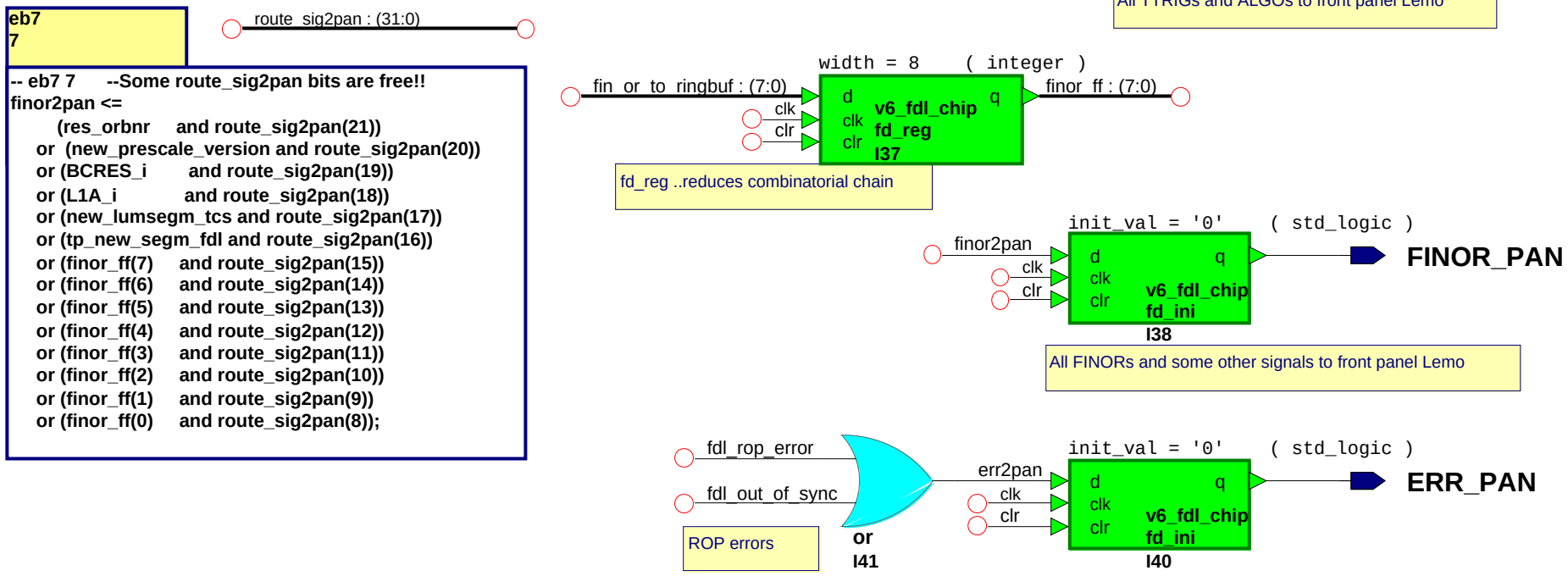
V100C:
Powerup values: en_chan links...
ROP: out_of_sync: <-- fifo_full, error <="0';

V100B:
ttrig 0-63: 16b presc, 20b ratecntrs
algo0-7: 20b presc, 24b ratecntrs
algo8-127: 18b presc, 24b ratecntrs
no_algo : 24b presc, 24b ratecntrs
Prescaler changed:
ROP record FFFF only at end.
common update pulse
orbit+lumin_seg cntrs: new resfresh selection
new_lum_seg_fdl/tcs-->testpoint


initial_value = X"00000DEB" ( std_logic_vector (31 DOWNTO 0) )
width = 16 ( integer )
vdin(15:0) : (31:0)
vme_en_common_reg(5) : (15:0)
clk
clr
vme_data_in : (width-1:0)
vme_wr_pulse
clk
clr
v6_vme
vme_wr_reg
ORBIT_LENGTH_REG
dout : (width-1:0)
orbit_length : (15:0)
eb1
1
-- eb1 1 STATUS bits
status2vme(31 downto 16) <= luminosity_nr;
status2vme(15 downto 12) <= X"0";
status2vme(11 downto 8) <=gt_status_i;
status2vme(7) <= fdl_bc_error;
status2vme(6) <= fdl_rop_error; -- =0..fixed
status2vme(5) <= fdl_out_of_sync;
status2vme(4) <= fdl_warn75;
status2vme(3) <= fdl_warn50;
status2vme(2) <= fdl_disconnected;
status2vme(1) <= fdl_busy;
status2vme(0) <= fdl_ready;
width = 32 ( integer )
vme_rd_status_reg(2) : (7:0)
luminosity_nr : (15:0)
vme_en
vme_data_out : (width-1:0)
v6_vme
vme_status_reg
FDL_STATUS
status : (width-1:0)
dout_fdl_status_reg : (31:0)
status2vme : (31:0)
initial_value = X"00000100" ( std_logic_vector (31 DOWNTO 0) )
width = 32 ( integer )
vdin : (31:0)
vme_en_common_reg(6) : (15:0)
clk
clr
vme_data_in : (width-1:0)
vme_wr_pulse
clk
clr
v6_vme
vme_wr_reg
ROUTE_SIG2PAN_REG1
dout : (width-1:0)
route_sig2pan : (31:0)
X"0100" ...show FinOR0
eb9
9
-- eb9 9
ld_new_lumsegm <= vme_en_common_reg(10);
initial_value = X"00100000" ( std_logic_vector (31 DOWNTO 0) )
width = 32 ( integer )
vdin : (31:0)
vme_en_common_reg(10) : (15:0)
clk
clr
vme_data_in : (width-1:0)
vme_wr_pulse
clk
clr
v6_vme
vme_wr_reg
LUM_SEGMPERIOD2
dout : (width-1:0)
lum_segm_period : (31:0)
X"0010_0000" = ~ 93 sec .. CMS value
X"000A4678" = ~60 sec ..previously
bit 15 ... 8: number of algos
bit 7 ... 0: number of ttrig
width = 16 ( integer )
vme_rd_status_reg(3) : (7:0)
slice_count : (15:0)
vme_en
vme_data_out : (width-1:0)
status : (width-1:0)
v6_vme
vme_status_reg
SLICE_COUNT_REG
dout_count_reg(15:0) : (31:0)
dout_count_reg(31:16) : (31:0)
eb5
5
slice_count( 7 downto 0) <= CONV_STD_LOGIC_VECTOR(no_of_ttrig, 8);
slice_count(15 downto 8) <= CONV_STD_LOGIC_VECTOR(no_of_algo,8);
init_val = X"00000000" ( std_logic_vector(31 DOWNTO 0) )
vdin : (31:0)
vme_en_common_reg(11) : (15:0)
apply_new_prescale_values
new_lumsegm_fdl
clk
clr
vme_data_in : (31:0)
vme_wr_pulse
apply_new_value
rate_cntr_update
clk
clear
v6_fdl_chip
slice_register
I26
prescale_reg : (31:0)
load_cntr
prescale_version : (31:0)
new_prescale_version
slice_register module used here
to update simultaneously to
prescale factors.
(2 reg's)
Prescale Version
V1011

# Fixed Number of ALGO (=128) and TTRIG (=64)



## Declarations

**Ports:**

```
apply_new_presc_value : std_logic
clear                 : std_logic
clk                   : std_logic
en_rate_cntrs         : std_logic
in_pad                : std_logic
refresh               : std_logic_vector(7 DOWNTO 0)
sel_sim               : std_logic
sim_spy               : std_logic
update_period         : std_logic_vector(7 DOWNTO 0)
vme_data_in           : std_logic_vector(31 downto 0)
vme_en_presc_value    : std_logic
vme_en_slice          : std_logic
rate_count            : std_logic_vector(31 DOWNTO 0)
to_ringbuff           : std_logic
to_sim_spy            : std_logic
trigger_bit           : std_logic_vector(7 DOWNTO 0)
veto_bit              : std_logic_vector(7 DOWNTO 0)
```

**Diagram Signals:**

```
SIGNAL bit_in               : std_logic
SIGNAL load_precntr         : std_logic
SIGNAL prescale_value       : std_logic_vector(31 DOWNTO 0)
SIGNAL prescaled_trigger    : std_logic
SIGNAL rate_cntr_update     : std_logic
SIGNAL rst_rate             : std_logic
SIGNAL slice_setup          : std_logic_vector(31 DOWNTO 0)
```

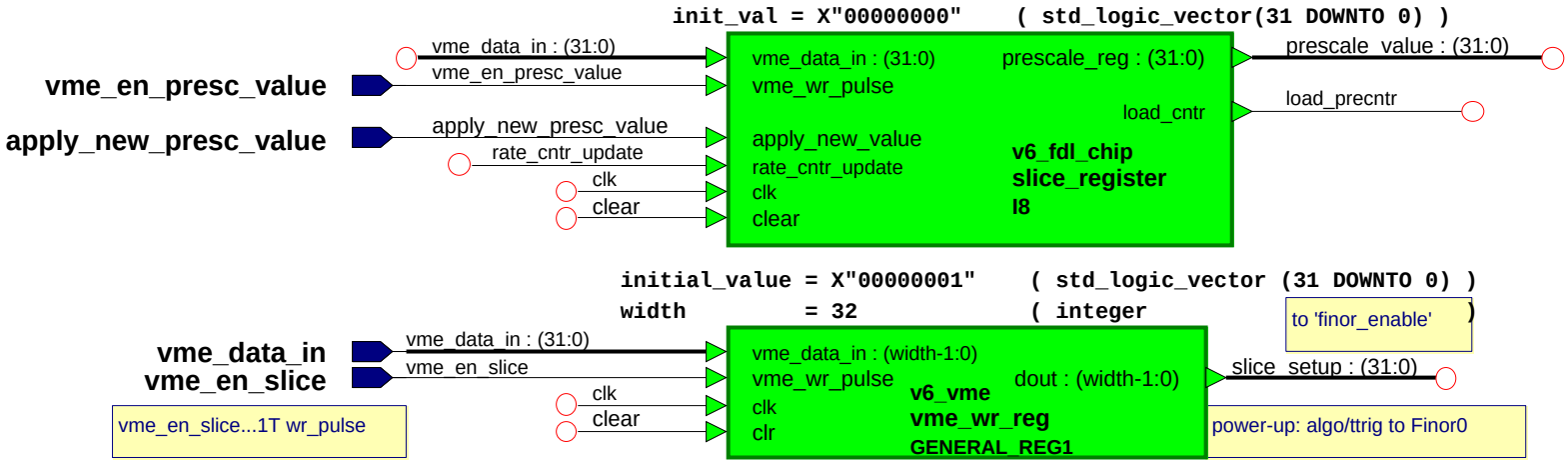
## Package List

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
USE ieee.numeric_std.all;
USE IEEE.VITAL_Timing.all;
LIBRARY unisim;
USE unisim.VCOMPONENTS.all;
```

load prescaler:
1. Load new prescale value into the register
2. Send the vme cmd pulse 'apply_new_presc_value' that loads the new value at the end of the luminosity segment into the pre_scaler.

power-up: X"0...00" = no prescaling

to 'prescaler'




| <company name> | | Project: | fdl_chip |
|---|---|---|---|
| | | V1011 simulated 12 Feb 2009 by AT | |
| Title: | <enter diagram title here> | | |
| Path: | v6_fdl_chip/fdl_slice/struct | | |
| Edited: | by taurok on 02 Sep 2009 | | |

## Declarations

### Ports:

```
algo                      : std_logic_vector(191 DOWNTO 0)
algo_s                    : std_logic_vector(5 DOWNTO 0)
apply_new_prescale_values : std_logic
bcres_int                 : std_logic
clear                     : std_logic
clk                       : std_logic
en_rate_cntrs             : std_logic
gen_reg                   : std_logic_vector(31 DOWNTO 0)
new_lumsegm_fdl           : std_logic
new_lumsegm_tcs           : std_logic
start_simspy_algo         : std_logic
start_simspy_finor        : std_logic
ttrig                     : std_logic_vector(63 DOWNTO 0)
update_step               : std_logic_vector(15 DOWNTO 0)
vme_addr                  : std_logic_vector(19 DOWNTO 1)
vme_data_in               : std_logic_vector(31 DOWNTO 0)
vme_en_finor_spy          : std_logic
vme_en_noalgo_presc       : std_logic
vme_en_noalgo_rate_reg    : std_logic
vme_en_noalgo_setup       : std_logic
vme_en_presc_value        : std_logic_vector(255 DOWNTO 0)
vme_en_sim_spy_mem        : std_logic_vector(7 DOWNTO 0)
vme_en_slice              : std_logic_vector(255 DOWNTO 0)
vme_en_slice_readback_mem : std_logic
vme_en_update_period_reg  : std_logic_vector(255 DOWNTO 0)
vme_rd_rate_reg           : std_logic_vector(255 DOWNTO 0)
vme_wr                    : std_logic
finor2ringbuf             : std_logic_vector(7 DOWNTO 0)
finor2tcs                 : std_logic_vector(7 DOWNTO 0)
noalgo2ringbuf            : std_logic
slice2panrouter           : std_logic_vector(255 DOWNTO 0)
to_ringbuff               : std_logic_vector(255 DOWNTO 0)
vme_data_out              : std_logic_vector(31 DOWNTO 0)
```

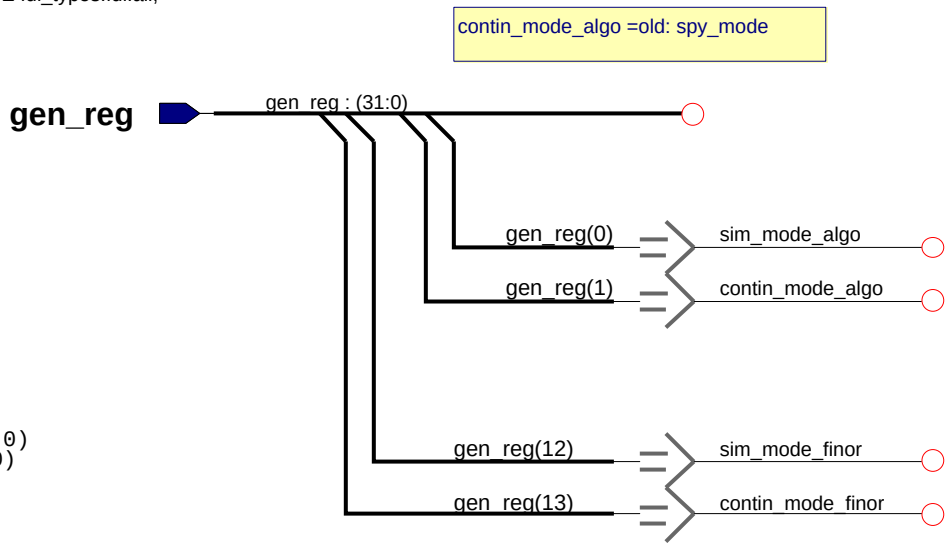
### Diagram Signals:

```
SIGNAL addr_cntr_finor          : std_logic_vector(11 DOWNTO 0)
SIGNAL addr_cntr_simspy         : std_logic_vector(11 DOWNTO 0)
SIGNAL algo_to_finor            : vec8_array(191 DOWNTO 0)
SIGNAL contin_mode_algo         : std_logic
SIGNAL contin_mode_finor        : std_logic
SIGNAL din1                     : std_logic_vector(31 DOWNTO 0)
SIGNAL din2                     : std_logic_vector(8 DOWNTO 0)
SIGNAL din_spy_mem              : std_logic_VECTOR(8 DOWNTO 0)
SIGNAL doutb                    : std_logic_VECTOR(no_of_ttrig-1 DOWNTO 0)
SIGNAL doutb1                   : std_logic_VECTOR(no_of_algo-1 DOWNTO 0)
SIGNAL en_mem_algo              : std_logic
SIGNAL en_mem_finor             : std_logic
SIGNAL finor_out                : std_logic_vector(7 downto 0)
SIGNAL ground2                  : std_logic_vector(8 DOWNTO 0)
SIGNAL ground_32                : std_logic_vector(31 DOWNTO 0)
SIGNAL high                     : std_logic
SIGNAL noalgo_rate_count        : std_logic_vector(31 DOWNTO 0)
SIGNAL noalgo_setup             : std_logic_vector(31 DOWNTO 0)
SIGNAL rate_cntr                : vec32_array(64 + no_of_algo -1 DOWNTO 0)
SIGNAL refresh                  : std_logic_vector(7 DOWNTO 0)
SIGNAL sim_finor                : std_logic_VECTOR(8 DOWNTO 0)
SIGNAL sim_mode_algo            : std_logic
SIGNAL sim_mode_finor           : std_logic
SIGNAL sim_spy                  : std_logic_vector(64 + no_of_algo -1 DOWNTO 0)
SIGNAL simspy0                  : std_logic_vector(15 DOWNTO 0)
SIGNAL simspy1                  : std_logic_vector(15 DOWNTO 0)
```

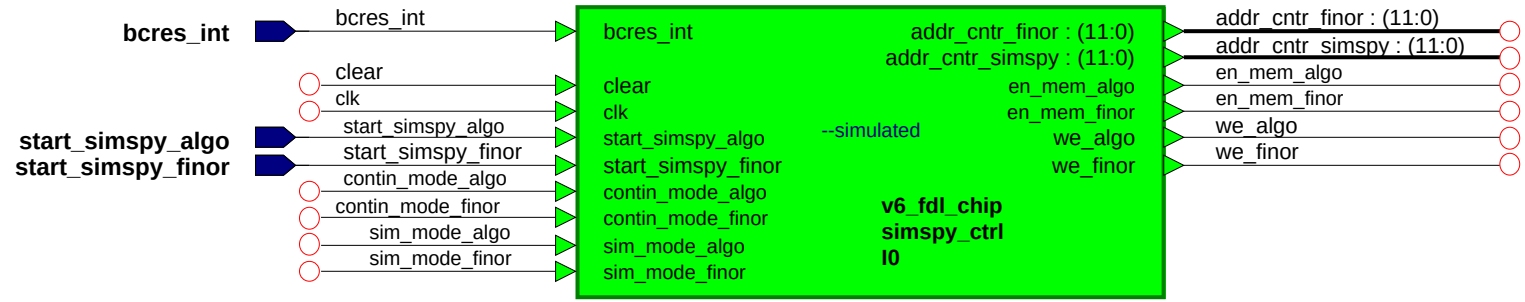
| <company name> | | Project: | fdl_chip |
|---|---|---|---|
| | | V1011 | |
| Title: | <enter diagram title here> | | |
| Path: | v6_fdl_chip/fdl_core/struct | | |
| Edited: | by taurok on 25 Sep 2009 | | |

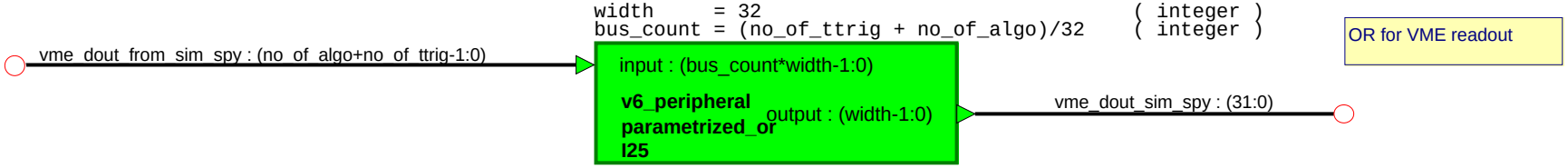
### Package List

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
USE ieee.numeric_std.all;
USE IEEE.VITAL_Timing.all;
LIBRARY UNISIM;
USE UNISIM.VPKG.all;
LIBRARY fdl_types;
USE fdl_types.fdl.all;
```



# SIM and SPY Controller without BCRES delay

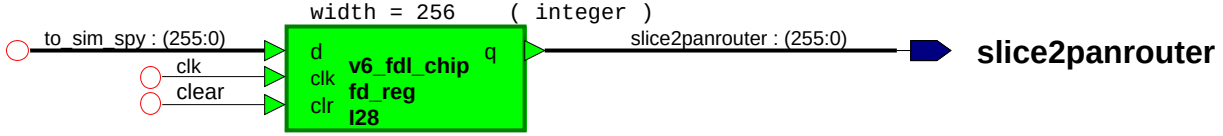



OR for VME readout

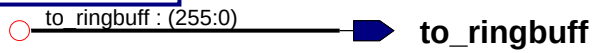
data are taken immediately after input flipflop
and sent to panel. fd_vec reduces combinatorial chain

# Slice to PANEL Router




**eb1 1**

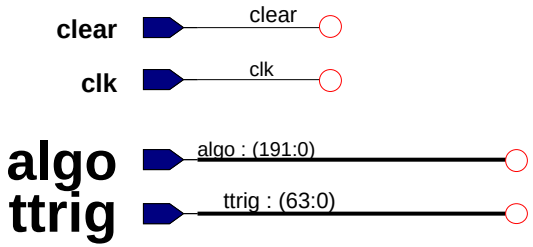
```
-- eb1 1
to_sim_spy(255 downto (no_of_algo+no_of_ttrig) ) <= (others => '0');
to_ringbuff(255 downto (no_of_algo+no_of_ttrig) ) <= (others => '0');
```



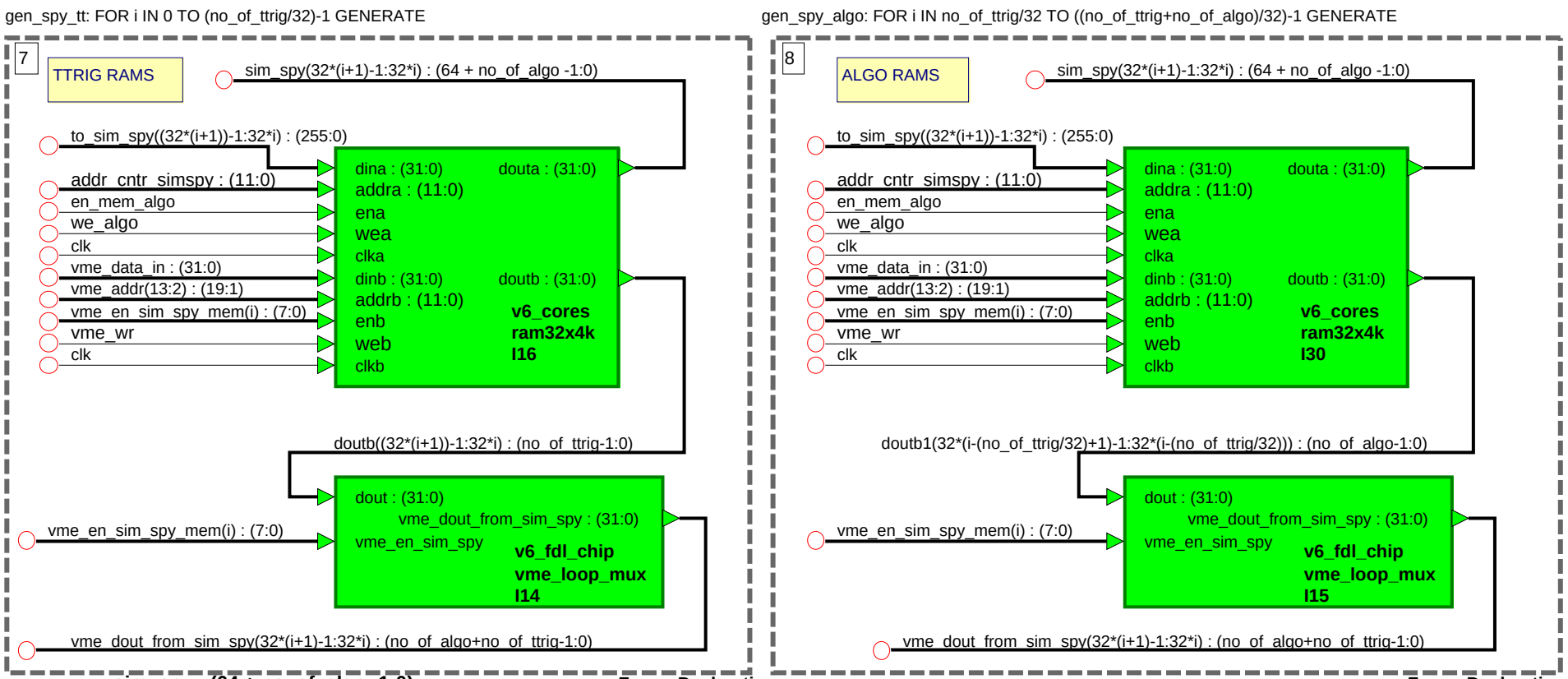

# to Ring Buffer

```
SIGNAL simspy10                 : std_logic_vector(15 DOWNTO 0)
SIGNAL simspy11                 : std_logic_vector(15 DOWNTO 0)
SIGNAL simspy2                  : std_logic_vector(15 DOWNTO 0)
SIGNAL simspy3                  : std_logic_vector(15 DOWNTO 0)
SIGNAL simspy4                  : std_logic_vector(15 DOWNTO 0)
SIGNAL simspy5                  : std_logic_vector(15 DOWNTO 0)
SIGNAL simspy6                  : std_logic_vector(15 DOWNTO 0)
SIGNAL simspy7                  : std_logic_vector(15 DOWNTO 0)
SIGNAL simspy8                  : std_logic_vector(15 DOWNTO 0)
SIGNAL simspy9                  : std_logic_vector(15 DOWNTO 0)
SIGNAL to_sim_spy               : std_logic_vector(255 DOWNTO 0)
SIGNAL trigger_bit_noalgo       : std_logic_vector(7 DOWNTO 0)
SIGNAL ttrig_to_finor           : vec8_array(63 DOWNTO 0)
SIGNAL update_per               : vec8_array(64 + no_of_algo -1 DOWNTO 0)
SIGNAL veto_to_finor            : vec8_array(63 DOWNTO 0)
SIGNAL vme_din                  : std_logic_vector(31 DOWNTO 0)
SIGNAL vme_dout_cntrs           : std_logic_vector(31 DOWNTO 0)
SIGNAL vme_dout_finor_spy_mem   : std_logic_vector(31 DOWNTO 0)
SIGNAL vme_dout_from_sim_spy    : std_logic_vector(no_of_algo+no_of_ttrig-1 DOWNTO 0)
SIGNAL vme_dout_noalgo_rate_reg : std_logic_vector(31 DOWNTO 0)
SIGNAL vme_dout_rate            : vec32_array(64 + no_of_algo -1 DOWNTO 0)
SIGNAL vme_dout_rdbk            : std_logic_VECTOR(31 DOWNTO 0)
SIGNAL vme_dout_sim_spy         : std_logic_vector(31 DOWNTO 0)
SIGNAL we_algo                  : std_logic
SIGNAL we_finor                 : std_logic
```

# Header



# SIM and SPY Memory




**sim_spy : (64 + no_of_algo -1:0)**

One 32x4k ram block is used for 16 slices.
Port A: SLICE
Port B: VME

**Frame Declarations**

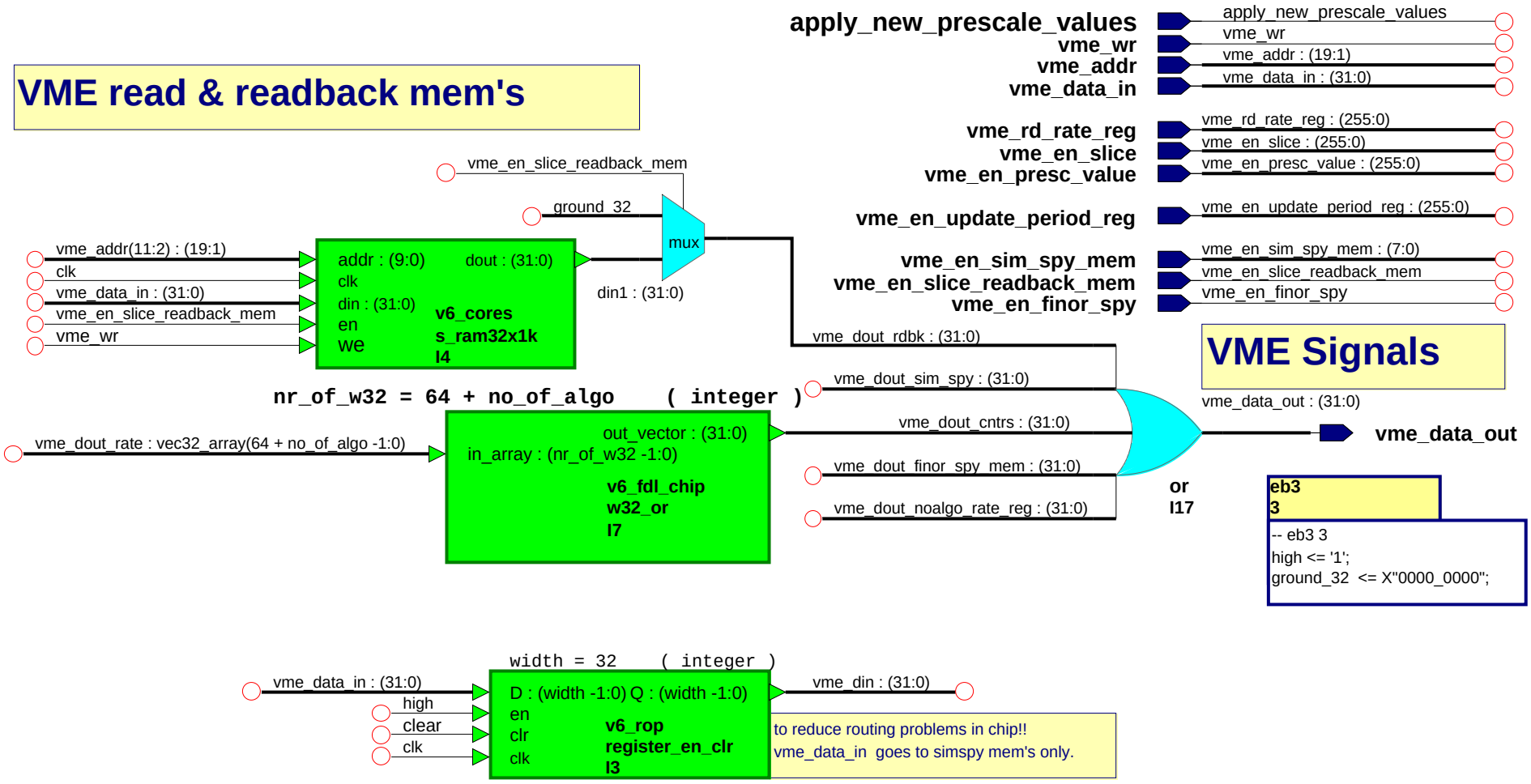
**eb2 2**

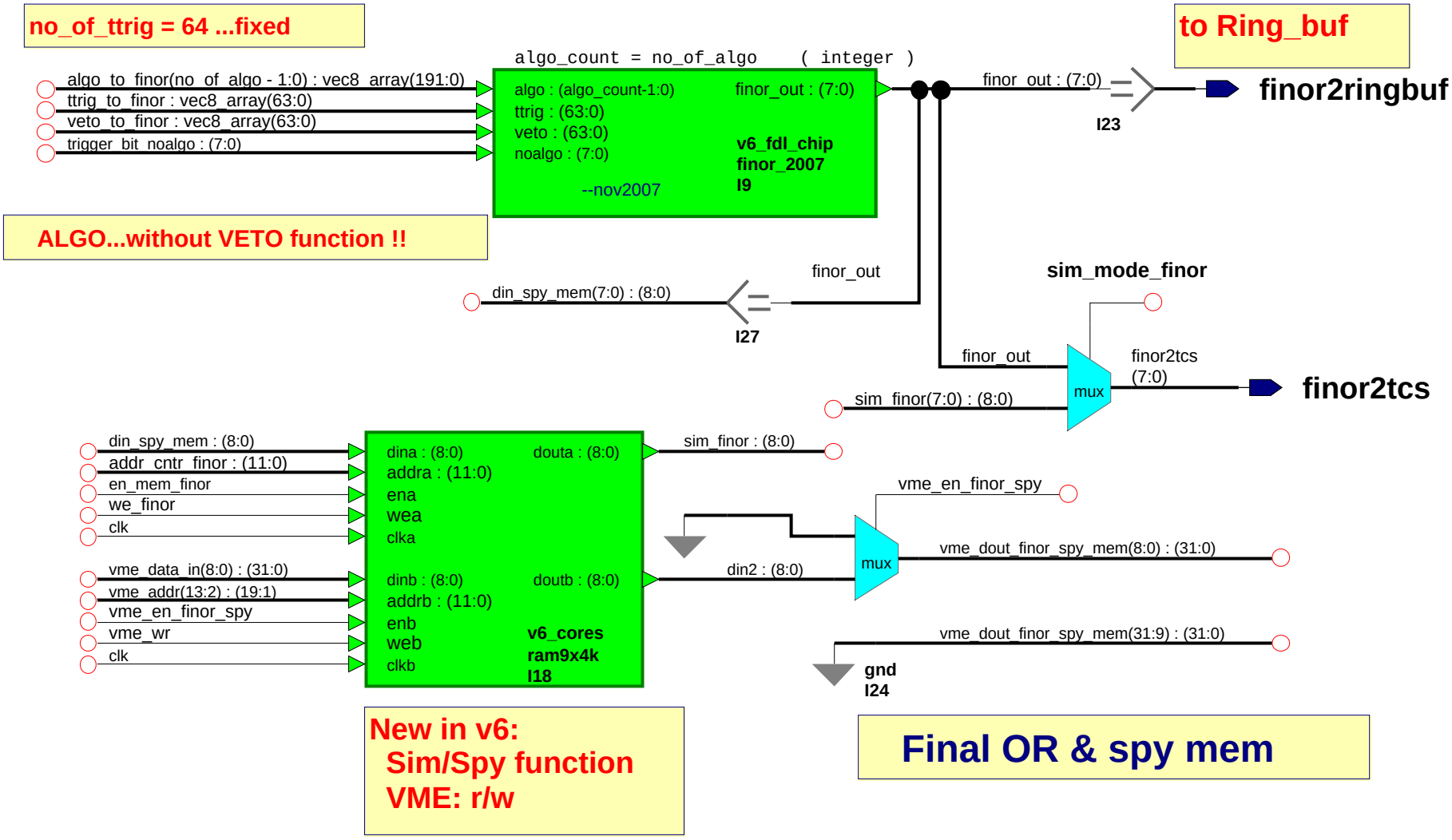
```
-- eb2 2 For SIMULATION ONLY !!!

simspy11 <= to_sim_spy(191 downto 176);
simspy10 <= to_sim_spy(175 downto 160);
simspy9 <= to_sim_spy(159 downto 144);
simspy8 <= to_sim_spy(143 downto 128);
simspy7 <= to_sim_spy(127 downto 112);
simspy6 <= to_sim_spy(111 downto 96);
simspy5 <= to_sim_spy(95 downto 80);
simspy4 <= to_sim_spy(79 downto 64);
simspy3 <= to_sim_spy(63 downto 48);
simspy2 <= to_sim_spy(47 downto 32);
simspy1 <= to_sim_spy(31 downto 16);
simspy0 <= to_sim_spy(15 downto 0);
```

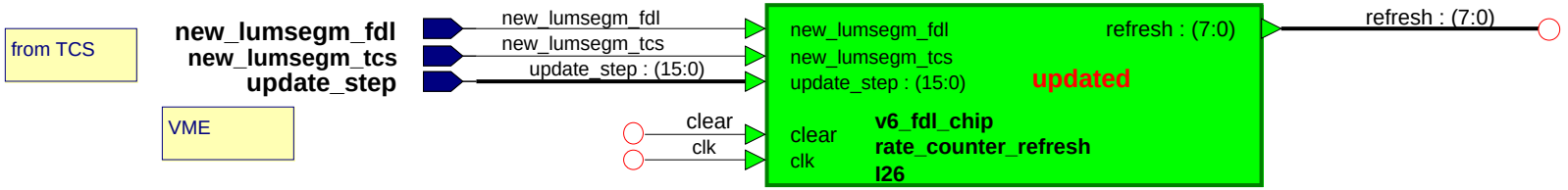
**Frame Declarations**

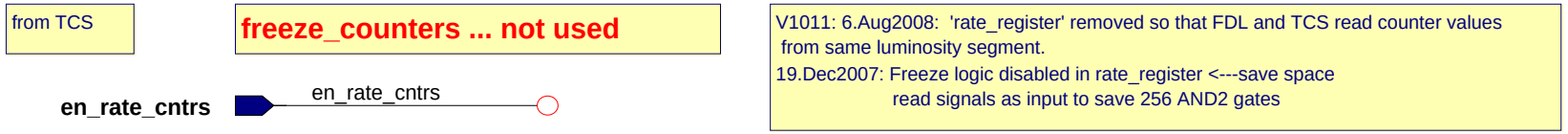
simspy11 : (15:0)
simspy10 : (15:0)
simspy9 : (15:0)
simspy8 : (15:0)
simspy7 : (15:0)
simspy6 : (15:0)
simspy5 : (15:0)
simspy4 : (15:0)
simspy3 : (15:0)
simspy2 : (15:0)
simspy1 : (15:0)
simspy0 : (15:0)

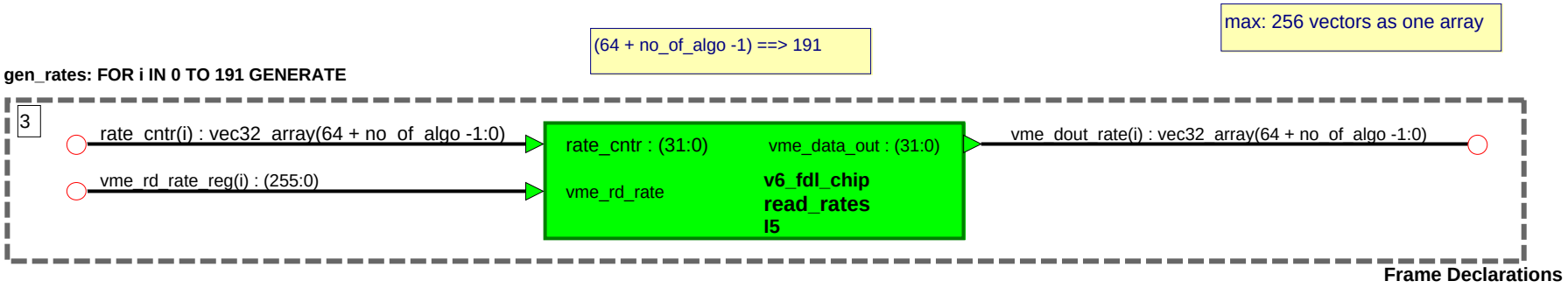

VME read & readback mem's
apply_new_prescale_values
vme_wr
vme_addr
vme_data_in
vme_rd_rate_reg
vme_en_slice
vme_en_presc_value
vme_en_update_period_reg
vme_en_sim_spy_mem
vme_en_slice_readback_mem
vme_en_finor_spy
apply_new_prescale_values
vme_wr
vme_addr : (19:1)
vme_data_in : (31:0)
vme_rd_rate_reg : (255:0)
vme_en_slice : (255:0)
vme_en_presc_value : (255:0)
vme_en_update_period_reg : (255:0)
vme_en_sim_spy_mem : (7:0)
vme_en_slice_readback_mem
vme_en_finor_spy
VME Signals
vme_en_slice_readback_mem
ground_32
mux
din1 : (31:0)
vme_addr(11:2) : (19:1)
clk
vme_data_in : (31:0)
vme_en_slice_readback_mem
vme_wr
addr : (9:0)
dout : (31:0)
clk
din : (31:0)
en
we
v6_cores
s_ram32x1k
I4
vme_dout_rdbk : (31:0)
vme_dout_sim_spy : (31:0)
vme_dout_cntrs : (31:0)
vme_dout_finor_spy_mem : (31:0)
vme_dout_noalgo_rate_reg : (31:0)
vme_data_out : (31:0)
vme_data_out
or
I17
nr_of_w32 = 64 + no_of_algo ( integer )
vme_dout_rate : vec32_array(64 + no_of_algo -1:0)
out_vector : (31:0)
in_array : (nr_of_w32 -1:0)
v6_fdl_chip
w32_or
I7
eb3
3
-- eb3 3
high <= '1';
ground_32 <= X"0000_0000";
width = 32 ( integer )
vme_data_in : (31:0)
high
clear
clk
D : (width -1:0)
Q : (width -1:0)
en
clr
clk
v6_rop
register_en_clr
I3
vme_din : (31:0)
to reduce routing problems in chip!!
vme_data_in goes to simspy mem's only.

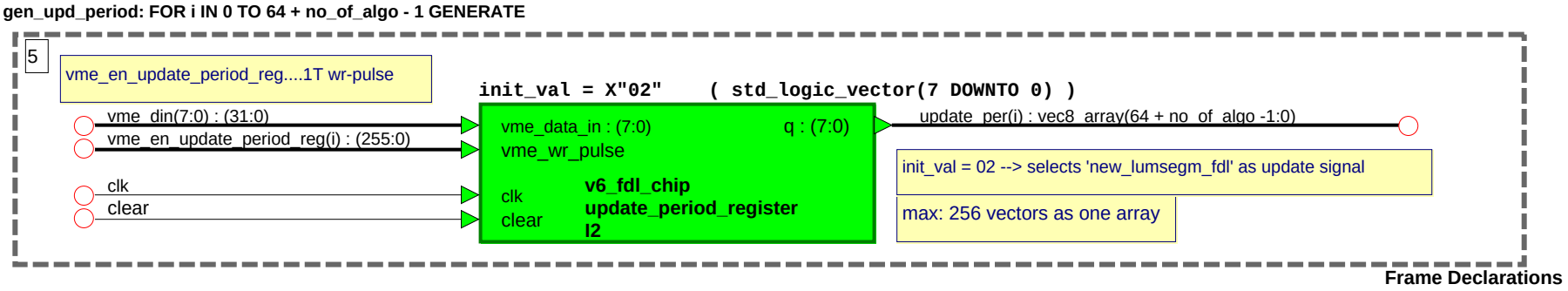



impl4: 102%: 18b pre/24b rate, 8algo full
impl3: 101%: 16b pre/24b rate, 8algo full


from TCS
new_lumsegm_fdl
new_lumsegm_tcs
update_step
VME
new_lumsegm_fdl
new_lumsegm_tcs
update_step : (15:0)
clear
clk
new_lumsegm_fdl
new_lumsegm_tcs
update_step : (15:0)
clear
clk
v6_fdl_chip
rate_counter_refresh
I26
refresh : (7:0)
updated
refresh : (7:0)

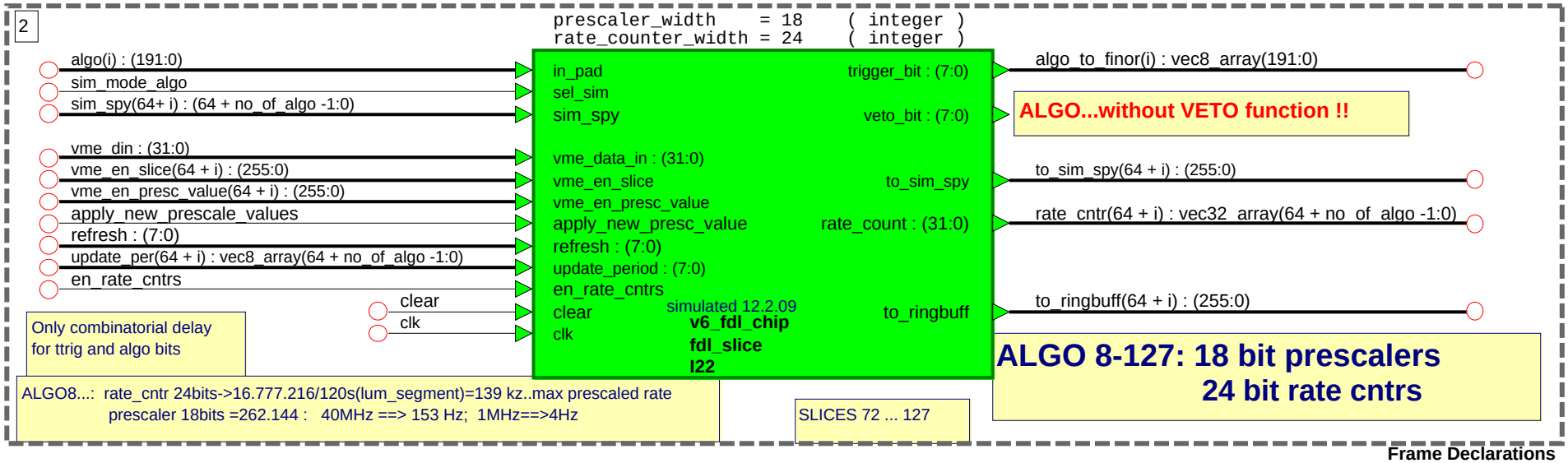

from TCS
freeze_counters ... not used
en_rate_cntrs
en_rate_cntrs
V1011: 6.Aug2008: 'rate_register' removed so that FDL and TCS read counter values
from same luminosity segment.
19.Dec2007: Freeze logic disabled in rate_register <---save space
read signals as input to save 256 AND2 gates


(64 + no_of_algo -1) ==> 191
max: 256 vectors as one array
gen_rates: FOR i IN 0 TO 191 GENERATE
3
rate_cntr(i) : vec32_array(64 + no_of_algo -1:0)
vme_rd_rate_reg(i) : (255:0)
rate_cntr : (31:0)
vme_data_out : (31:0)
vme_rd_rate
v6_fdl_chip
read_rates
I5
vme_dout_rate(i) : vec32_array(64 + no_of_algo -1:0)
Frame Declarations


gen_upd_period: FOR i IN 0 TO 64 + no_of_algo - 1 GENERATE
5
vme_en_update_period_reg....1T wr-pulse
init_val = X"02" ( std_logic_vector(7 DOWNTO 0) )
vme_din(7:0) : (31:0)
vme_en_update_period_reg(i) : (255:0)
clk
clear
vme_data_in : (7:0)
vme_wr_pulse
clk
clear
v6_fdl_chip
update_period_register
I2
q : (7:0)
update_per(i) : vec8_array(64 + no_of_algo -1:0)
init_val = 02 --> selects 'new_lumsegm_fdl' as update signal
max: 256 vectors as one array
Frame Declarations

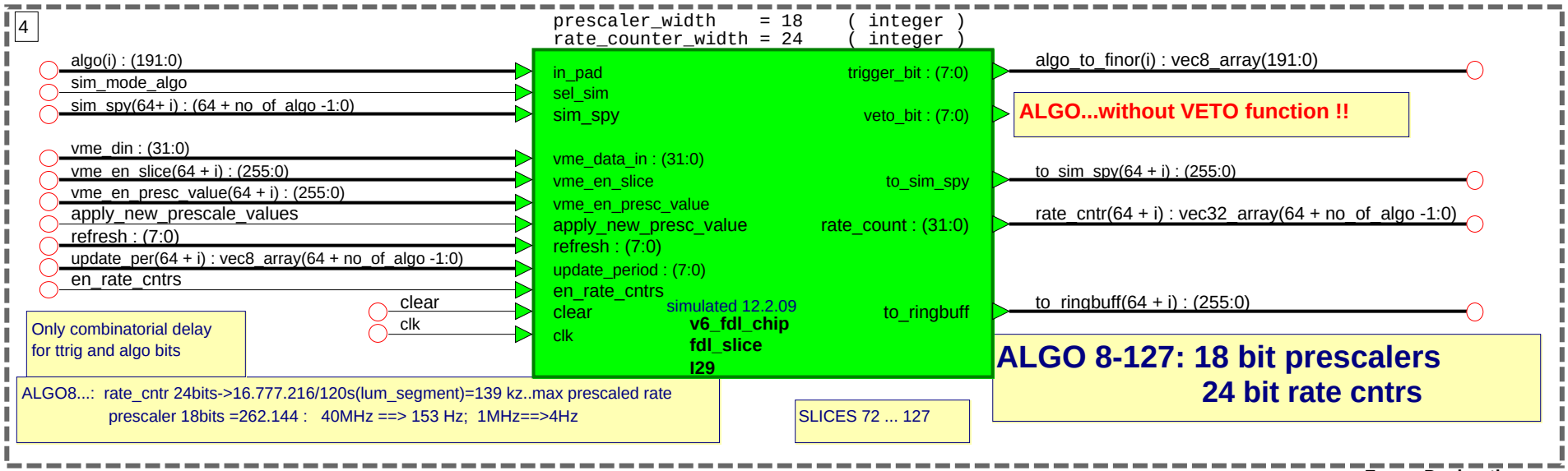
18 bit prescaler: 1 MHz trigger/(2**18) = 4 Hz trigger
24 bit prescaler: 40 MHz trigger/(2**24) = 2 Hz trigger

**gen_algo8_127: FOR i IN 64 TO no_of_algo-1 GENERATE**




Frame Declarations

**gen_algo8_128: FOR i IN 8 TO 63 GENERATE**

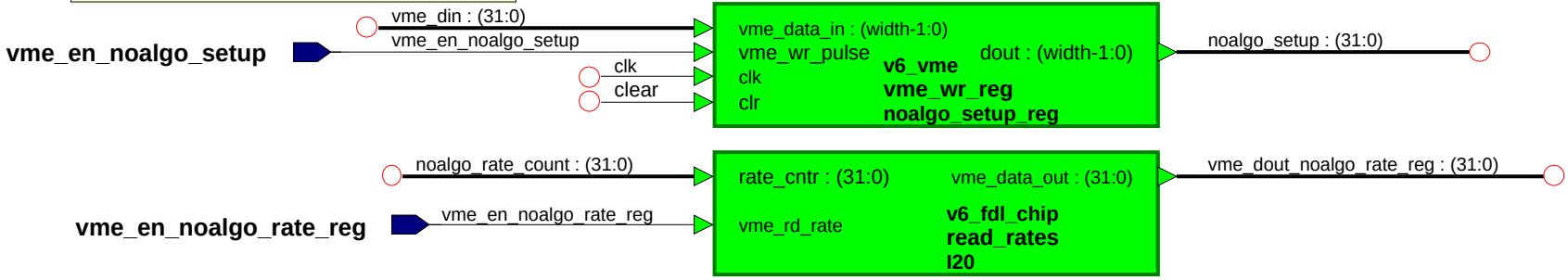


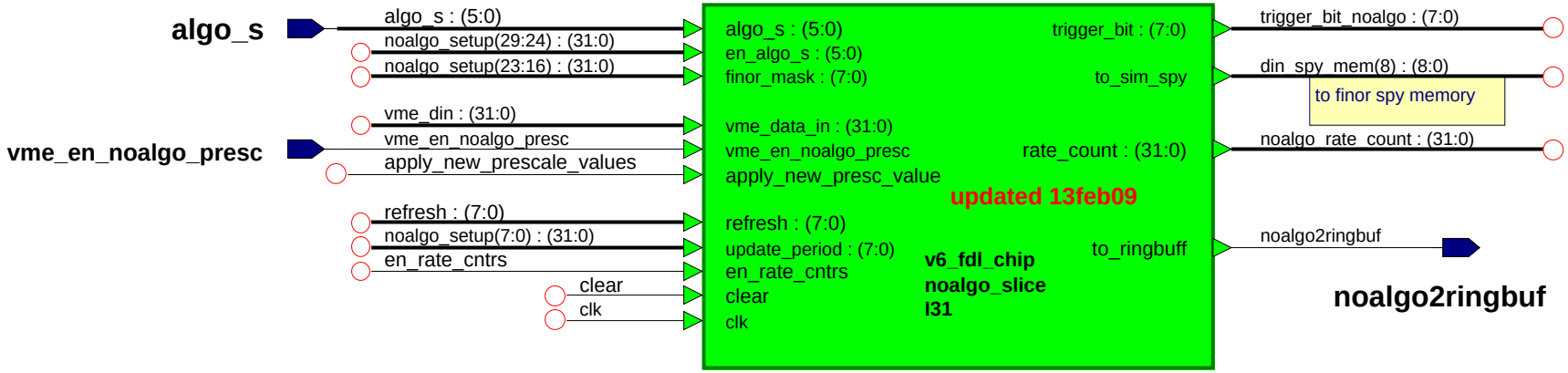

Frame Declarations

# VME Registers

31-14:en_algo_mask =0F..include algo_s 0...3
23-16: finor_mask = 00
15-8: =00
7-0: update_period = 02..new_lum_segm_fdl

```
initial_value = X"0F000002"    ( std_logic_vector (31 DOWNTO 0) )
width         = 32             ( integer                        )
```



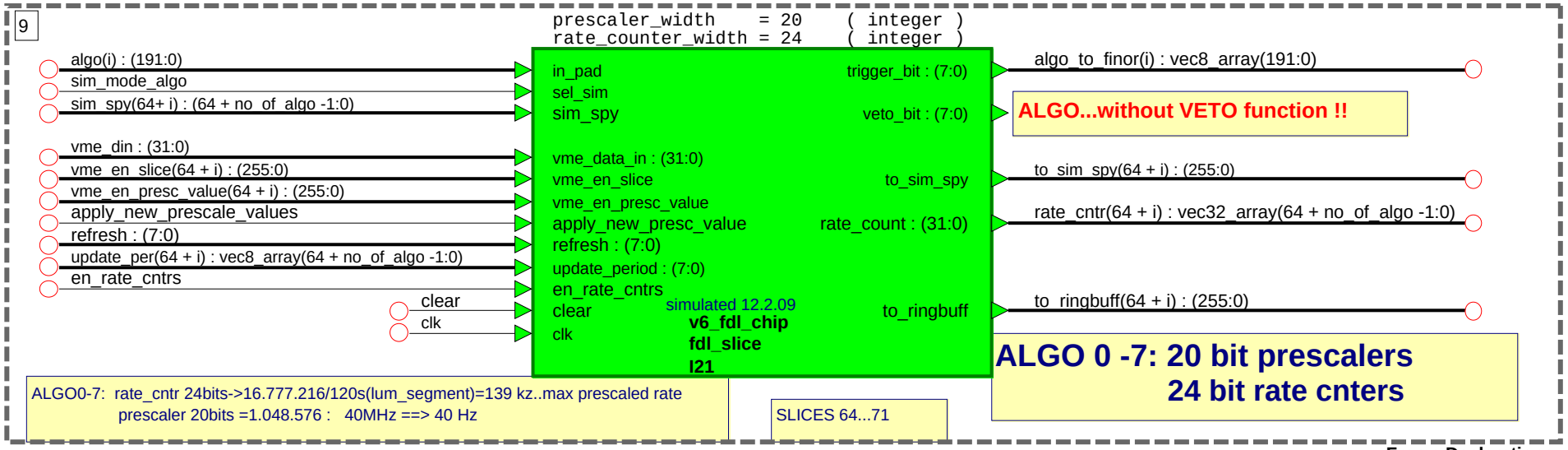




NO_ALGO: rate_cntr 24bits->16.777.216/120s(lum_segment)=139 kz..max prescaled rate
prescaler 24bits =16.777.216 : 40MHz ==> 2,4 Hz

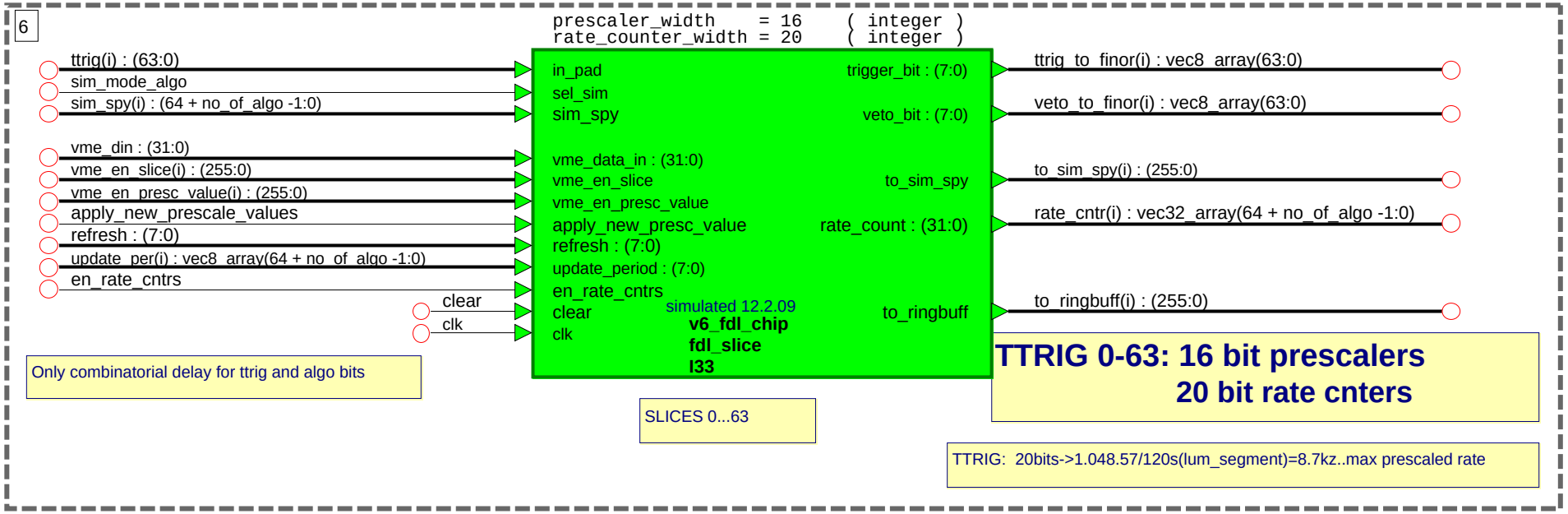
**24 bit prescaler**
**24 bit rate counter**

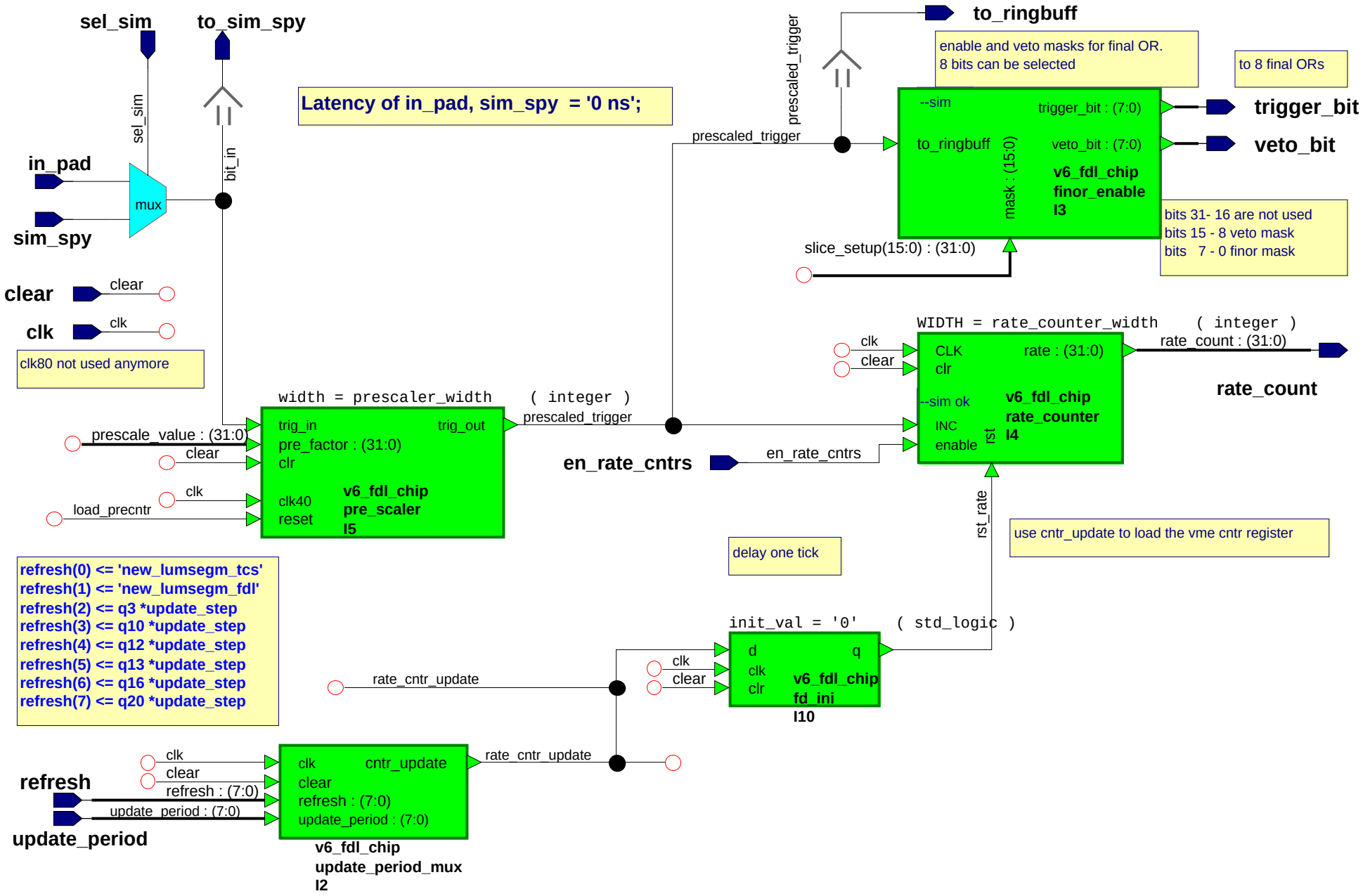
# NOALGO Slice

**gen_algo0_7: FOR i IN 0 TO 7 GENERATE**




**gen_ttrig0_63: FOR i IN 0 TO 63 GENERATE**




sel_sim
to_sim_spy
sel_sim
bit_in
in_pad
mux
sim_spy
Latency of in_pad, sim_spy = '0 ns';
prescaled_trigger
to_ringbuff
enable and veto masks for final OR.
8 bits can be selected
to 8 final ORs
--sim
trigger_bit : (7:0)
to_ringbuff
veto_bit : (7:0)
mask : (15:0)
v6_fdl_chip
finor_enable
I3
trigger_bit
veto_bit
bits 31- 16 are not used
bits 15 - 8 veto mask
bits 7 - 0 finor mask
slice_setup(15:0) : (31:0)
clear
clear
clk
clk
clk80 not used anymore
WIDTH = rate_counter_width ( integer )
rate_count : (31:0)
CLK
rate : (31:0)
clr
--sim ok
v6_fdl_chip
rate_counter
INC
enable
rst
I4
rate_count
width = prescaler_width ( integer )
trig_in
trig_out
prescale_value : (31:0)
pre_factor : (31:0)
clear
clr
clk
load_precntr
clk40
reset
v6_fdl_chip
pre_scaler
I5
prescaled_trigger
en_rate_cntrs
en_rate_cntrs
rst_rate
use cntr_update to load the vme cntr register
delay one tick
refresh(0) <= 'new_lumsegm_tcs'
refresh(1) <= 'new_lumsegm_fdl'
refresh(2) <= q3 *update_step
refresh(3) <= q10 *update_step
refresh(4) <= q12 *update_step
refresh(5) <= q13 *update_step
refresh(6) <= q16 *update_step
refresh(7) <= q20 *update_step
init_val = '0' ( std_logic )
d
q
clk
clr
v6_fdl_chip
fd_ini
I10
rate_cntr_update
clk
clear
rate_cntr_update
clk
clear
refresh : (7:0)
update_period : (7:0)
refresh
update_period
clk
cntr_update
clear
refresh : (7:0)
update_period : (7:0)
v6_fdl_chip
update_period_mux
I2

```
-- Procedures:
-- 1.) New PRESCALE FACTORS
--   a)  Load a new prescale factor into 'vme_reg' by vme-software.
--   b) The vme common command pulse 'apply_new_presc_value' sets the request_ff =1.
--   c) The next 'rate_cntr_update' signal ( default:= 'new_luminosity_segment' pulse from FDL)
--     ==> clears the request_ff
--     ==> clears the pre_counter (to restart with the new prescale factor)
--     ==> moves the new prescale factor from the 'vme_reg' into the 'prescale_reg'
-- 2.)   Prescaling
--     a) The next trig signals increment the pre_counter until the
--     new prescale value and the counter are equal (limit =1).
--     b) The next trigger passes through (en_trigger=1),
--         ==>  increments the rate counter and
--         ==>  clears the pre_counter again.
--       //Loop: 2a==>2b==>2a==>2b==> ...
-- 3.)   Rate counter readout:
--       Every 'rate_cntr_update''  signal ( default:= 'new_luminosity_segment' pulse from FDL)
--       one tick later:
--        ==> moves the actual rate counter value into a rate_register and
--        ==> clears the rate counter.
--
-- *** The update is done for all Algo and Technical Trigger bits concurrently. ***
```

**Global Actions**
**Pre Actions:**
**Post Actions:**

**Concurrent Statements**

**Architecture Declarations**

**Signals Status**

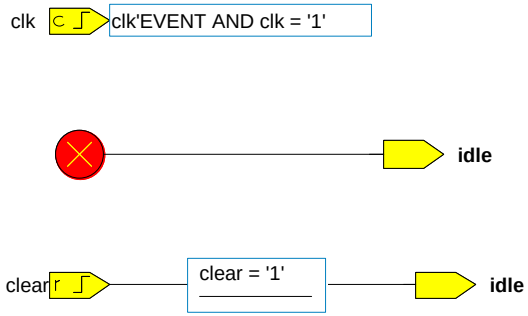
| SIGNAL | MODE | DEFAULT | RESET | SCHEME |
|---|---|---|---|---|
| inc_bx_in_ev_ctr | OUT | '0' | | COMB |
| read_fifo | OUT | '0' | | COMB |
| res_bx_in_ev_ctr | OUT | '0' | | COMB |
| mode_5bx | LOCAL | | '0' | CLKD |

**State Register Statements**

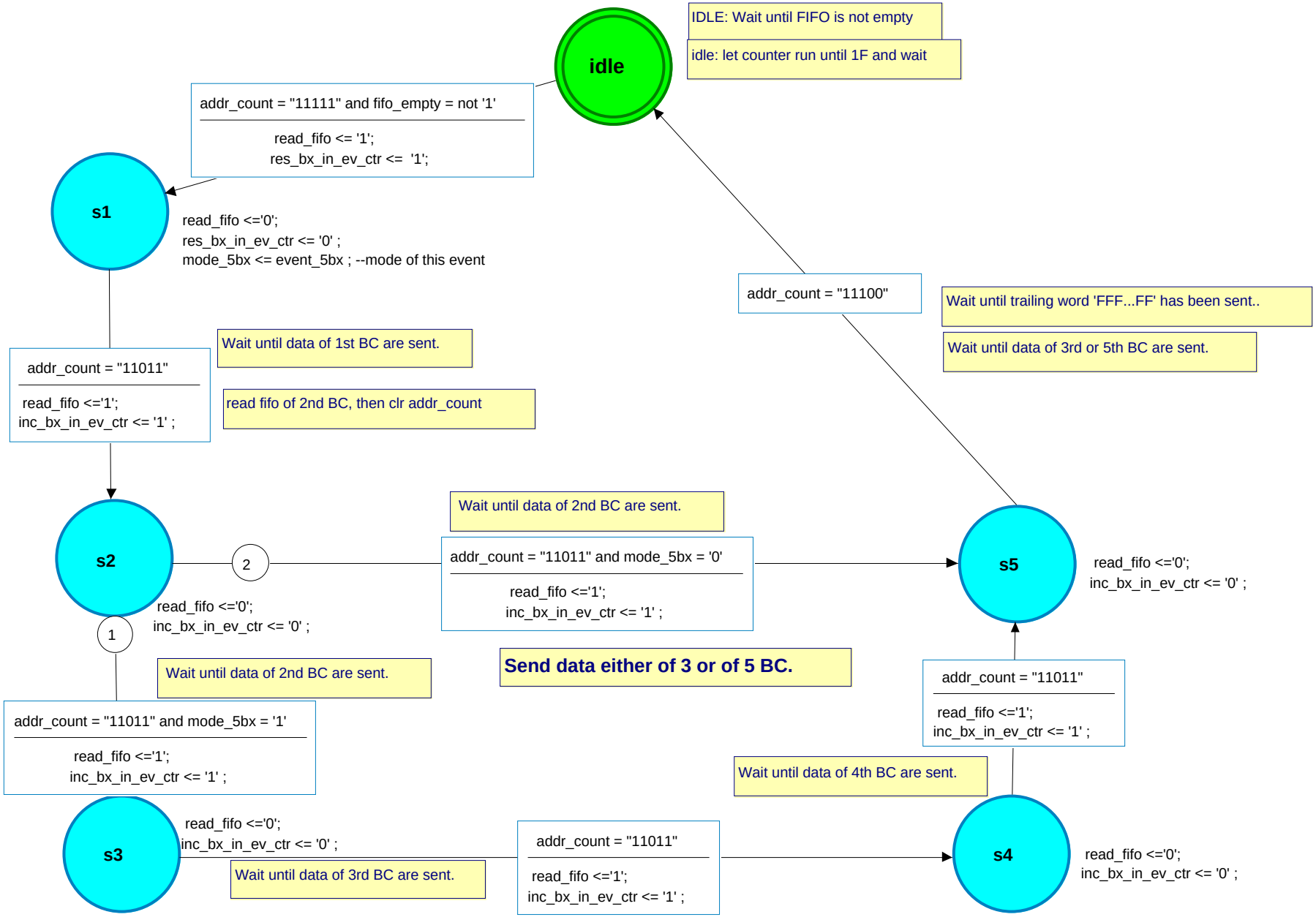
**Process Declarations**
**Clocked Process:**
**Output Process:**

**Package List**

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
```




| <company name> | | Project: | fdl_chip |
|---|---|---|---|
| Title: | <enter diagram title here> | 28 words per bx --> 7 W64 per bx | |
| Path: | v6_rop/ROC/fsm | | |
| Edited: | by taurok on 08 Sep 2009 | | |


idle
IDLE: Wait until FIFO is not empty
idle: let counter run until 1F and wait
addr_count = "11111" and fifo_empty = not '1'
read_fifo <= '1';
res_bx_in_ev_ctr <= '1';
s1
read_fifo <='0';
res_bx_in_ev_ctr <= '0' ;
mode_5bx <= event_5bx ; --mode of this event
addr_count = "11011"
read_fifo <='1';
inc_bx_in_ev_ctr <= '1' ;
Wait until data of 1st BC are sent.
read fifo of 2nd BC, then clr addr_count
s2
2
1
read_fifo <='0';
inc_bx_in_ev_ctr <= '0' ;
Wait until data of 2nd BC are sent.
addr_count = "11011" and mode_5bx = '0'
read_fifo <='1';
inc_bx_in_ev_ctr <= '1' ;
Send data either of 3 or of 5 BC.
addr_count = "11011" and mode_5bx = '1'
read_fifo <='1';
inc_bx_in_ev_ctr <= '1' ;
s3
read_fifo <='0';
inc_bx_in_ev_ctr <= '0' ;
Wait until data of 3rd BC are sent.
addr_count = "11011"
read_fifo <='1';
inc_bx_in_ev_ctr <= '1' ;
s4
read_fifo <='0';
inc_bx_in_ev_ctr <= '0' ;
Wait until data of 4th BC are sent.
addr_count = "11011"
read_fifo <='1';
inc_bx_in_ev_ctr <= '1' ;
s5
read_fifo <='0';
inc_bx_in_ev_ctr <= '0' ;
Wait until data of 2nd BC are sent.
addr_count = "11100"
Wait until trailing word 'FFF...FF' has been sent..
Wait until data of 3rd or 5th BC are sent.

**Declarations**

**Ports:**

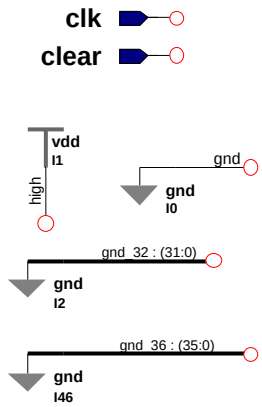
```
BCres            : std_logic
BOARD_ID         : std_logic_vector(15 DOWNTO 0)
IDLE_CODE        : std_logic_vector(27 DOWNTO 0)
L1A              : std_logic
L1Res            : std_logic
ORBIT_LENGTH     : std_logic_vector(11 DOWNTO 0)
clear            : std_logic
clk              : std_logic
finor2ringbuf    : std_logic_vector(7 DOWNTO 0)
hardres          : std_logic
latency_delay    : std_logic_vector(15 DOWNTO 0)
lum_segm_nr      : std_logic_vector(15 DOWNTO 0)
noalgo2ringbuf   : std_logic
orbit_nr         : std_logic_vector(31 DOWNTO 0)
prescale_version : std_logic_vector(31 DOWNTO 0)
res_evnr         : std_logic
sel_long         : std_logic
to_ringbuf       : std_logic_vector(255 DOWNTO 0)
chan_link_rec    : std_logic_vector(27 DOWNTO 0)
error            : std_logic
out_of_sync      : STD_ULOGIC
warn50           : std_logic
warn75           : std_logic
```

**Diagram Signals:**

```
SIGNAL FINOR_DAT        : std_logic_vector(15 downto 0)
SIGNAL SEL              : std_logic_vector(4 DOWNTO 0)
SIGNAL addr_count       : std_logic_vector(4 DOWNTO 0)
SIGNAL bc_to_ereg       : std_logic_vector(15 DOWNTO 0)
SIGNAL bx_in_event      : std_logic_vector(3 DOWNTO 0)
SIGNAL bx_nr_eword      : std_logic_vector(15 DOWNTO 0)
SIGNAL c_word           : std_logic_vector(15 DOWNTO 0)
SIGNAL data_count       : std_logic_vector(1 DOWNTO 0)
SIGNAL dina             : std_logic_VECTOR(35 DOWNTO 0)
SIGNAL doutb            : std_logic_VECTOR(no_of_algo-1 DOWNTO 0)
SIGNAL doutb1           : std_logic_VECTOR(no_of_ttrig-1 DOWNTO 0)
SIGNAL event_5bx        : std_logic -- equal mode_5bx
SIGNAL event_cntnr      : std_logic_vector(31 DOWNTO 0)
SIGNAL event_nr         : std_logic_vector(31 DOWNTO 0)
SIGNAL event_nr_fifo    : std_logic_vector(31 DOWNTO 0)
SIGNAL fifo_empty       : std_logic
SIGNAL fifo_full        : std_logic
SIGNAL fifo_init        : std_logic
SIGNAL finor_ringb      : std_logic_VECTOR(35 DOWNTO 0)
SIGNAL finor_ringb2fifo : std_logic_VECTOR(35 DOWNTO 0)
SIGNAL gnd              : std_logic
SIGNAL gnd_32           : std_logic_vector(31 DOWNTO 0)
SIGNAL gnd_36           : std_logic_vector(35 DOWNTO 0)
SIGNAL high             : std_logic
SIGNAL inc_bx_in_ev_ctr : std_logic
SIGNAL inc_l1a_addr     : std_logic
SIGNAL l1a_addr         : std_logic_vector(31 DOWNTO 0)
SIGNAL ld_l1a_addr      : std_logic
SIGNAL lum_segm_nr_fifo : std_logic_vector(31 DOWNTO 0)
SIGNAL lum_segment_nr   : std_logic_vector(15 DOWNTO 0)
SIGNAL lumsegm2ringb    : std_logic_VECTOR(31 DOWNTO 0)
SIGNAL lumsegm_ringb    : std_logic_vector(31 DOWNTO 0)
SIGNAL no_l1a_waiting   : std_logic
SIGNAL orb_nr_ringb     : std_logic_VECTOR(31 DOWNTO 0)
SIGNAL orbit_nr_fifo    : std_logic_vector(31 DOWNTO 0)
SIGNAL orbitnr          : std_logic_vector(31 DOWNTO 0)
SIGNAL rd_addr          : std_logic_vector(11 DOWNTO 0)
SIGNAL rd_addr_cntr     : std_logic_vector(11 DOWNTO 0)
SIGNAL rd_addr_cntr32b  : std_logic_vector(31 DOWNTO 0)
SIGNAL rd_fifo          : std_logic
SIGNAL rd_l1a_fifo      : std_logic
SIGNAL readout_data     : std_logic_vector((no_of_algo + no_of_ttrig) -1 DOWNTO 0)
SIGNAL readout_ffifo    : std_logic_vector(35 DOWNTO 0)
SIGNAL reg2mux          : std_logic_vector(256 DOWNTO 0)
SIGNAL res_bx_in_ev_ctr : std_logic
SIGNAL res_rd_addr      : STD_ULOGIC
SIGNAL res_wr_addr      : std_logic
SIGNAL store_fifo_data  : std_logic
SIGNAL to_c             : std_logic_vector(15 DOWNTO 0)
SIGNAL to_finorreg      : std_logic_vector(15 DOWNTO 0)
SIGNAL wr_addr          : std_logic_vector(11 DOWNTO 0)
SIGNAL wr_fifo          : std_logic
SIGNAL wr_fifo1         : std_logic
```

**Package List**

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
USE IEEE.VITAL_Timing.all;
LIBRARY UNISIM;
USE UNISIM.VPKG.all;
USE ieee.numeric_std.all;
```




| | | | |
|---|---|---|---|
| <company name> | | Project: | fdl_chip |
| | | V1015: I1a_sm inserted. | |
| Title: | <enter diagram title here> | Time between triggers can be 0 ...n bx. | |
| Path: | v6_rop/rop/struct | | |
| Edited: | by taurok on 14 Sep 2009 | | |

# Ringbuffers

# FIFOs

**to_ringbuf** to_ringbuf : (255:0)

g2: FOR i IN no_of_ttrig/32 TO (no_of_ttrig+no_of_algo)/32-1 GENERATE

3 **ALGO Ring Buffers 32x512**

wr_addr(8:0) : (11:0)
rd_addr(8:0) : (11:0)
clk
clk
to_ringbuf(32*(i+1)-1:32*i) : (255:0)
gnd_32 : (31:0)
high
high
high
gnd

addra : (8:0) douta : (31:0)
addrb : (8:0) doutb : (31:0)
clka
clkb
dina : (31:0)
dinb : (31:0)
ena
enb **v6_cores**
wea **ringbuf32x512**
web **I29**

doutb(32*(i-no_of_ttrig/32+1)-1:32*(i-no_of_ttrig/32)) : (no_of_algo-1:0)

clk
fifo_init
wr_fifo
rd_fifo

clk dout : (31:0)
sinit data_count : (1:0)
din : (31:0) full
wr_en empty
rd_en **v6_cores**
**fifo32x512**
**I16**

readout_data(32*(i+1)-1:32*i) : ((no_of_algo + no_of_ttrig) -1:0)

'empty' = std_logic_vector ( ((no_of_algo + no_of_ttrig)/32) -1 downto 0 );
-- 192/32 -1 = 6-1=5 //// 256/32 -1 = 8-1 =7

**Frame Declarations**

g6: FOR i IN 0 TO no_of_ttrig/32-1 GENERATE

7 **Technical Trigger Ring Buffers 32x512**

wr_addr(8:0) : (11:0)
rd_addr(8:0) : (11:0)
clk
clk
to_ringbuf(32*(i+1)-1:32*i) : (255:0)
gnd_32 : (31:0)
high
high
high
gnd

addra : (8:0) douta : (31:0)
addrb : (8:0) doutb : (31:0)
clka
clkb
dina : (31:0)
dinb : (31:0)
ena
enb **v6_cores**
wea **ringbuf32x512**
web **I33**

doutb1(32*(i+1)-1:32*i) : (no_of_ttrig-1:0)

clk
fifo_init
wr_fifo
rd_fifo

clk dout : (31:0)
sinit data_count : (1:0)
din : (31:0) full
wr_en empty
rd_en **v6_cores**
**fifo32x512**
**I17**

readout_data(32*(i+1)-1:32*i) : ((no_of_algo + no_of_ttrig) -1:0)

**Frame Declarations**

```
-- eb5 5
dina(7 downto 0)<= finor2ringbuf;
dina(8)          <= noalgo2ringbuf;
dina(11 downto 9) <= (others =>'0');
dina(23 downto 12) <= wr_addr;
dina(35 downto 24) <=(others =>'0');
```

**finor2ringbuf : (7:0)**
wr_addr : (11:0)
**noalgo2ringbuf**

**eb6**
**6**

rd_addr : (11:0)
clk
clear

width = 12 ( integer )

d q
clk **v6_fdl_chip**
clr **fd_reg**
**I32**

finor_ringb2fifo(35:24) : (35:0)

wr_addr(8:0) : (11:0)
rd_addr(8:0) : (11:0)
clk
clk
dina : (35:0)
gnd_36 : (35:0)
high
high
high
gnd

addra : (8:0) douta : (35:0)
addrb : (8:0) doutb : (35:0)
clka
clkb
dina : (35:0)
dinb : (35:0)
ena
enb **v6_cores**
wea **ringbuf36x512**
web **I38**

finor_ringb : (35:0)

**eb7**
**7**

```
finor_ringb2fifo(23 downto 12) <= finor_ringb(23 downto 12);
finor_ringb2fifo(11)           <= event_5bx;
finor_ringb2fifo(10)           <= '0';
finor_ringb2fifo(9)            <= '0';
finor_ringb2fifo(8 downto 0) <= finor_ringb(8 downto 0);
```

**wr_fifo ...from L1A State Machine**
**rd_addr ... from Readout Counter**

clk
fifo_init
finor_ringb2fifo : (35:0)
wr_fifo
rd_fifo

clk dout : (35:0)
sinit full
din : (35:0) empty
wr_en data_count : (1:0)
rd_en **v6_cores**
**fifo36x512**
**I8**

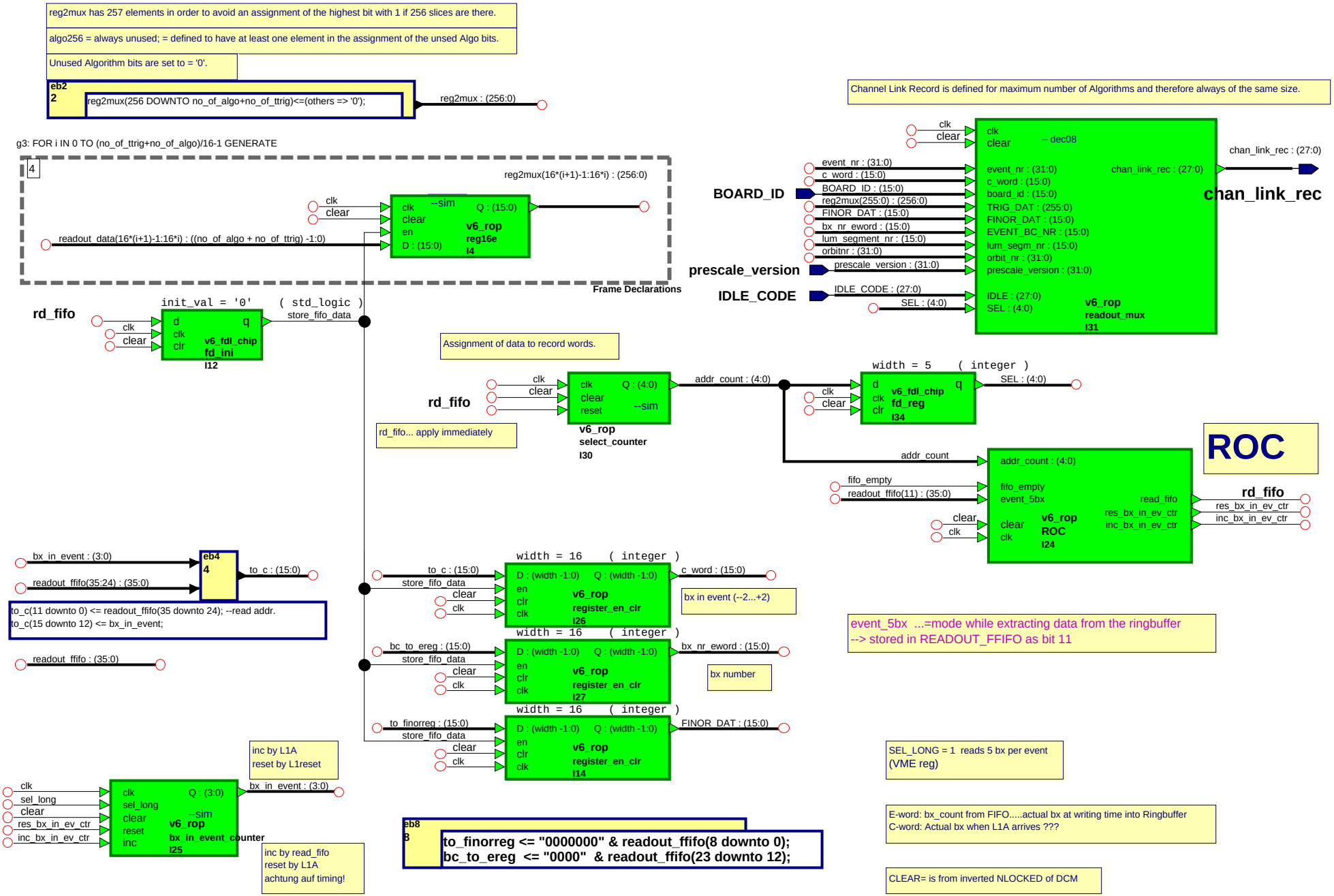
readout_ffifo : (35:0)
fifo_full
fifo_empty
data_count : (1:0)

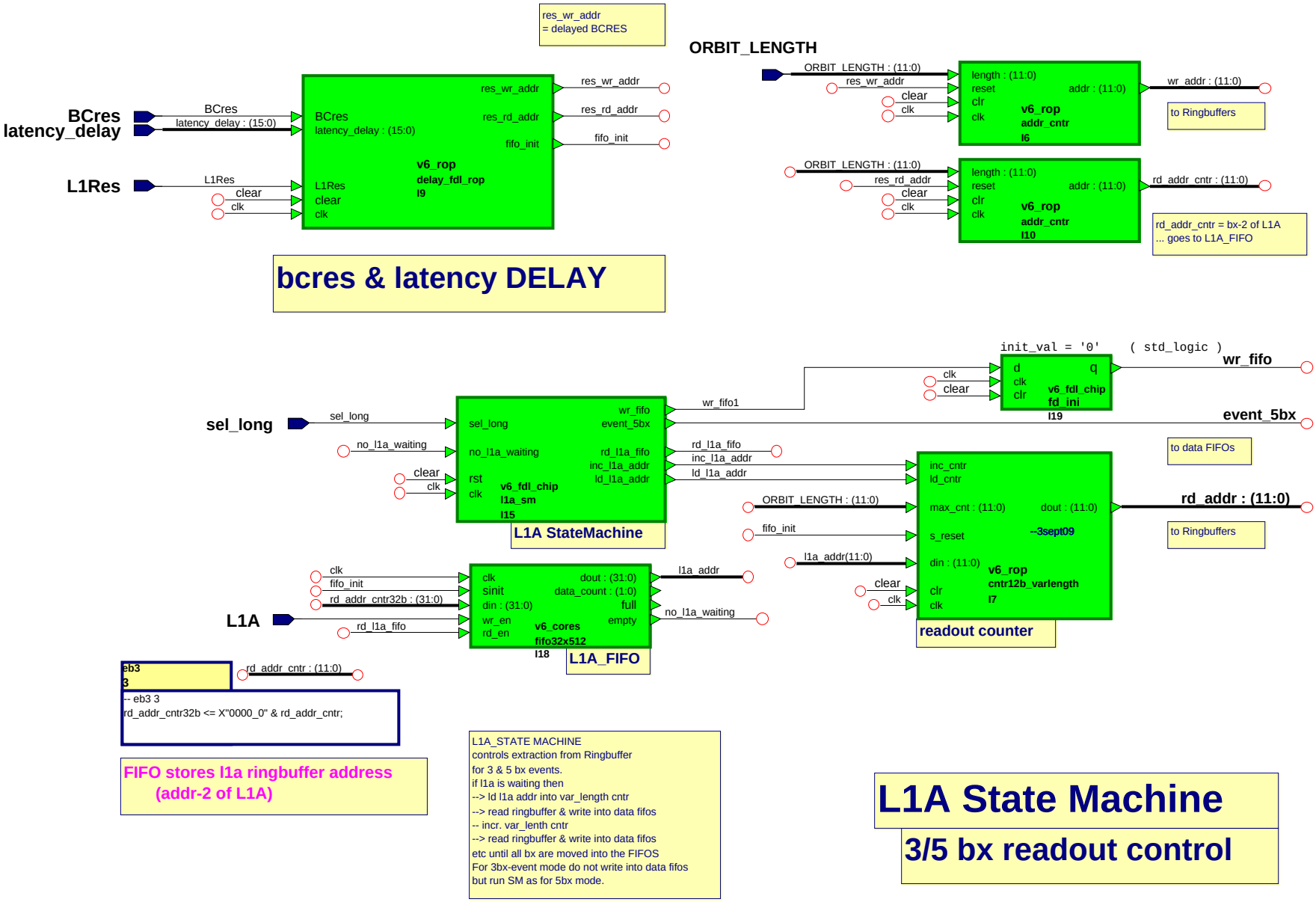
fifo_empty ==> ROC state machine

data_count ==> 'WARNING'
fifo_full ==> 'OUT_OF_SYNC'

FIFO content:
Bit 35-24 : bc nr, read_address of Ringbuf ==>C-word
Bit 23-12 : bc nr, when writing into Ringbuffer ==>E-word
Bit 11 : event_5bx ==> ROC state machine
Bit10 : '0'
Bit9 : '0'
Bit 8 : noalgo ==> DATA word
Bit 7-0 : finor ==> DATA word

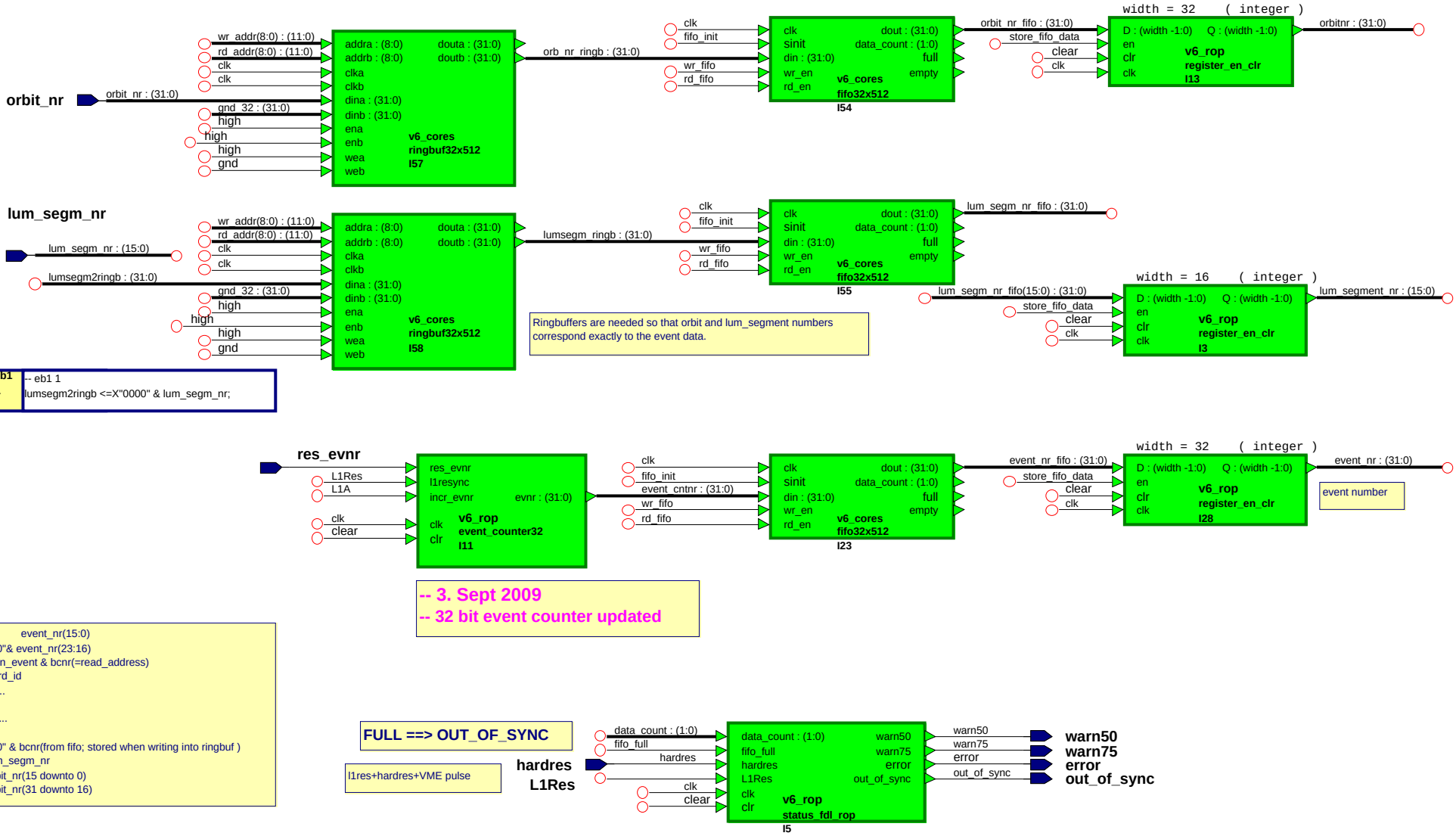
event_5bx ...=mode while extracting data from the ringbuffer


reg2mux has 257 elements in order to avoid an assignment of the highest bit with 1 if 256 slices are there.
algo256 = always unused; = defined to have at least one element in the assignment of the unsed Algo bits.
Unused Algorithm bits are set to = '0'.
eb2
2
reg2mux(256 DOWNTO no_of_algo+no_of_ttrig)<=(others => '0');
reg2mux : (256:0)
Channel Link Record is defined for maximum number of Algorithms and therefore always of the same size.
g3: FOR i IN 0 TO (no_of_ttrig+no_of_algo)/16-1 GENERATE
4
reg2mux(16*(i+1)-1:16*i) : (256:0)
readout_data(16*(i+1)-1:16*i) : ((no_of_algo + no_of_ttrig) -1:0)
clk
clear
--sim
Q : (15:0)
v6_rop
reg16e
I4
Frame Declarations
BOARD_ID
prescale_version
IDLE_CODE
event_nr : (31:0)
c_word : (15:0)
BOARD_ID : (15:0)
reg2mux(255:0) : (256:0)
FINOR_DAT : (15:0)
bx_nr_eword : (15:0)
lum_segment_nr : (15:0)
orbitnr : (31:0)
prescale_version : (31:0)
IDLE_CODE : (27:0)
SEL : (4:0)
-- dec08
chan_link_rec : (27:0)
TRIG_DAT : (255:0)
EVENT_BC_NR : (15:0)
lum_segm_nr : (15:0)
orbit_nr : (31:0)
IDLE : (27:0)
v6_rop
readout_mux
I31
chan_link_rec
rd_fifo
init_val = '0'
( std_logic )
store_fifo_data
d
q
clk
clr
v6_fdl_chip
fd_ini
I12
Assignment of data to record words.
rd_fifo
Q : (4:0)
reset
addr_count : (4:0)
v6_rop
select_counter
I30
rd_fifo... apply immediately
width = 5
( integer )
fd_reg
I34
SEL : (4:0)
ROC
addr_count
fifo_empty
readout_ffifo(11) : (35:0)
event_5bx
read_fifo
res_bx_in_ev_ctr
inc_bx_in_ev_ctr
ROC
I24
bx_in_event : (3:0)
readout_ffifo(35:24) : (35:0)
eb4
4
to_c : (15:0)
to_c(11 downto 0) <= readout_ffifo(35 downto 24); --read addr.
to_c(15 downto 12) <= bx_in_event;
readout_ffifo : (35:0)
width = 16
( integer )
D : (width -1:0)
Q : (width -1:0)
en
clr
clk
v6_rop
register_en_clr
I26
c_word : (15:0)
bx in event (--2...+2)
bc_to_ereg : (15:0)
I27
bx_nr_eword : (15:0)
bx number
to_finorreg : (15:0)
I14
FINOR_DAT : (15:0)
event_5bx ...=mode while extracting data from the ringbuffer
--> stored in READOUT_FFIFO as bit 11
sel_long
res_bx_in_ev_ctr
inc_bx_in_ev_ctr
inc
v6_rop
bx_in_event_counter
I25
bx_in_event : (3:0)
inc by L1A
reset by L1reset
inc by read_fifo
reset by L1A
achtung auf timing!
eb8
8
to_finorreg <= "0000000" & readout_ffifo(8 downto 0);
bc_to_ereg <= "0000" & readout_ffifo(23 downto 12);
SEL_LONG = 1 reads 5 bx per event
(VME reg)
E-word: bx_count from FIFO.....actual bx at writing time into Ringbuffer
C-word: Actual bx when L1A arrives ???
CLEAR= is from inverted NLOCKED of DCM


res_wr_addr
= delayed BCRES
ORBIT_LENGTH
BCres
latency_delay
L1Res
BCres
latency_delay : (15:0)
L1Res
clear
clk
res_wr_addr
res_rd_addr
fifo_init
v6_rop
delay_fdl_rop
I9
ORBIT_LENGTH : (11:0)
res_wr_addr
clear
clk
length : (11:0)
reset
clr
clk
addr : (11:0)
v6_rop
addr_cntr
I6
wr_addr : (11:0)
to Ringbuffers
ORBIT_LENGTH : (11:0)
res_rd_addr
clear
clk
length : (11:0)
reset
clr
clk
addr : (11:0)
v6_rop
addr_cntr
I10
rd_addr_cntr : (11:0)
rd_addr_cntr = bx-2 of L1A
... goes to L1A_FIFO
bcres & latency DELAY
init_val = '0' ( std_logic )
wr_fifo
clk
clear
d
clk
clr
q
v6_fdl_chip
fd_ini
I19
wr_fifo1
event_5bx
to data FIFOs
sel_long
sel_long
no_l1a_waiting
clear
clk
sel_long
no_l1a_waiting
rst
clk
v6_fdl_chip
l1a_sm
I15
wr_fifo
event_5bx
rd_l1a_fifo
inc_l1a_addr
ld_l1a_addr
L1A StateMachine
rd_l1a_fifo
inc_l1a_addr
ld_l1a_addr
inc_cntr
ld_cntr
ORBIT_LENGTH : (11:0)
fifo_init
l1a_addr(11:0)
clear
clk
max_cnt : (11:0)
s_reset
din : (11:0)
clr
clk
dout : (11:0)
--3sept09
v6_rop
cntr12b_varlength
I7
rd_addr : (11:0)
to Ringbuffers
readout counter
clk
fifo_init
rd_addr_cntr32b : (31:0)
L1A
rd_l1a_fifo
clk
sinit
din : (31:0)
wr_en
rd_en
dout : (31:0)
data_count : (1:0)
full
empty
v6_cores
fifo32x512
I18
l1a_addr
no_l1a_waiting
L1A_FIFO
eb3
3
rd_addr_cntr : (11:0)
-- eb3 3
rd_addr_cntr32b <= X"0000_0" & rd_addr_cntr;
FIFO stores l1a ringbuffer address
(addr-2 of L1A)
L1A_STATE MACHINE
controls extraction from Ringbuffer
for 3 & 5 bx events.
if l1a is waiting then
--> ld l1a addr into var_length cntr
--> read ringbuffer & write into data fifos
-- incr. var_lenth cntr
--> read ringbuffer & write into data fifos
etc until all bx are moved into the FIFOS
For 3bx-event mode do not write into data fifos
but run SM as for 5bx mode.
L1A State Machine
3/5 bx readout control

# Ringbuffer & FIFO for Event-, Orbit- and Luminosity Segment number V1011

wr_fifo ...from L1A State Machine
rd_addr ... from Readout Counter




-- 3. Sept 2009
-- 32 bit event counter updated

A: event_nr(15:0)
B: X"00"& event_nr(23:16)
C: bx_in_event & bcnr(=read_address)
D: board_id
1: ttrig...

1: algo...

E0: X"0" & bcnr(from fifo; stored when writing into ringbuf )
E1: lum_segm_nr
E2: orbit_nr(15 downto 0)
E3: orbit_nr(31 downto 16)

**Package List**

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE IEEE.VITAL_Timing.all;
USE IEEE.STD_LOGIC_UNSIGNED.all;
LIBRARY UNISIM;
USE UNISIM.VPKG.all;

**Declarations**

**Ports:**

```
BCres         : std_logic
L1Res         : std_logic
clear         : std_logic
clk           : std_logic
latency_delay : std_logic_vector(15 DOWNTO 0)
sel_long      : std_logic
fifo_init     : std_logic
res_rd_addr   : STD_ULOGIC
res_wr_addr   : std_logic
```

**Diagram Signals:**

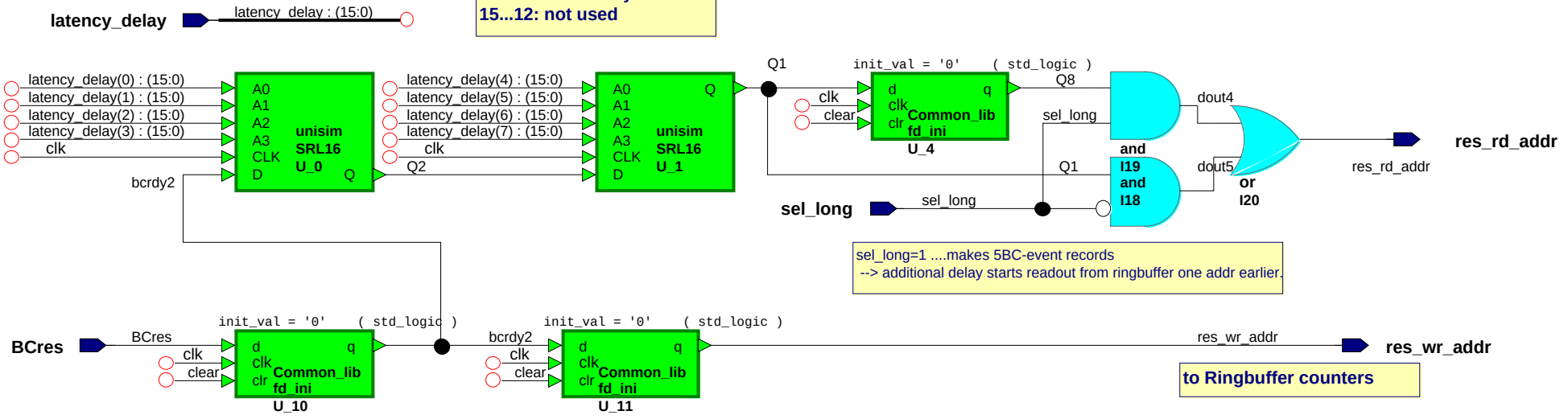
```
SIGNAL Q1           : STD_ULOGIC
SIGNAL Q2           : STD_ULOGIC
SIGNAL Q8           : STD_ULOGIC
SIGNAL bcrdy2       : std_logic
SIGNAL dout4        : STD_ULOGIC
SIGNAL dout5        : std_logic
SIGNAL q10          : std_logic
SIGNAL q11          : std_logic
SIGNAL q12          : std_logic
SIGNAL q13          : std_logic
SIGNAL q9           : std_logic
```

| | | | |
|---|---|---|---|
| <company name> | | Project: | fdl_chip |
| | | <enter comments here> | |
| Title: | <enter diagram title here> | | |
| Path: | v6_rop/delay_fdl_rop/struct | | |
| Edited: | by taurok on 28 Mai 2009 | | |

IF bcres_delay(=lat_delay(11:8)) = 0 and lat_delay(7:0) =0 then 'res_wr_addr' is 1 tick earlier then 'res_rd_addr',
to avoid writing and reading from same address at the same clock cycle.
NEW: bcres_delay preceeds latency_delay(7:0) !!!

Minimum Delay =2
-Analog to IN chips on GMT

**Latency delay register:**
**3...0: Latency delay 1**
**7...4: Latency delay 2**
**11...8: BCRes delay 1**
**15...12: not used**




sel_long=1 ....makes 5BC-event records
--> additional delay starts readout from ringbuffer one addr earlier.

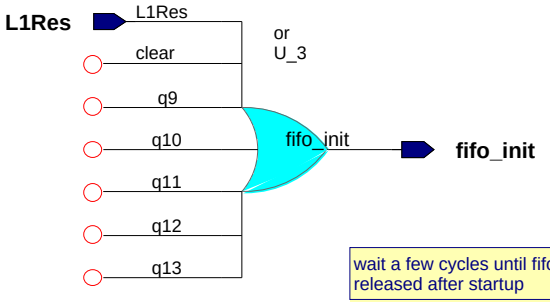



**to Ringbuffer counters**

**2 FD..to save data concurrently with SPY memory**

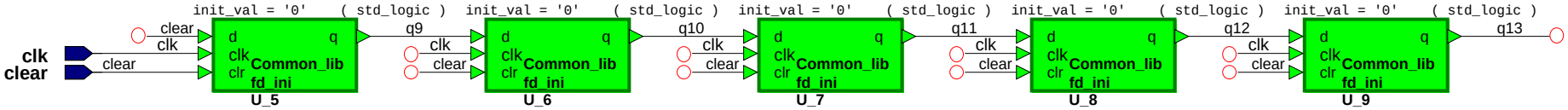
The read pointer(address) is at least one location behind the write pointer.

**V10014: BCres is delayed in "time_run_control"**




clear = inverted LOCKED from DCM
(used after startup)

wait a few cycles until fifos are
released after startup



**Global Actions**
**Pre Actions:**
**Post Actions:**

**Concurrent Statements**
--stored with event data and
-- used by ROC state machine
event_5bx <= mode_5bx ;

**Architecture Declarations**

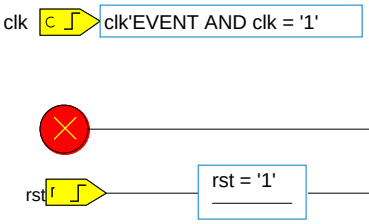
**State Register Statements**

**Process Declarations**
**Clocked Process:**
**Output Process:**

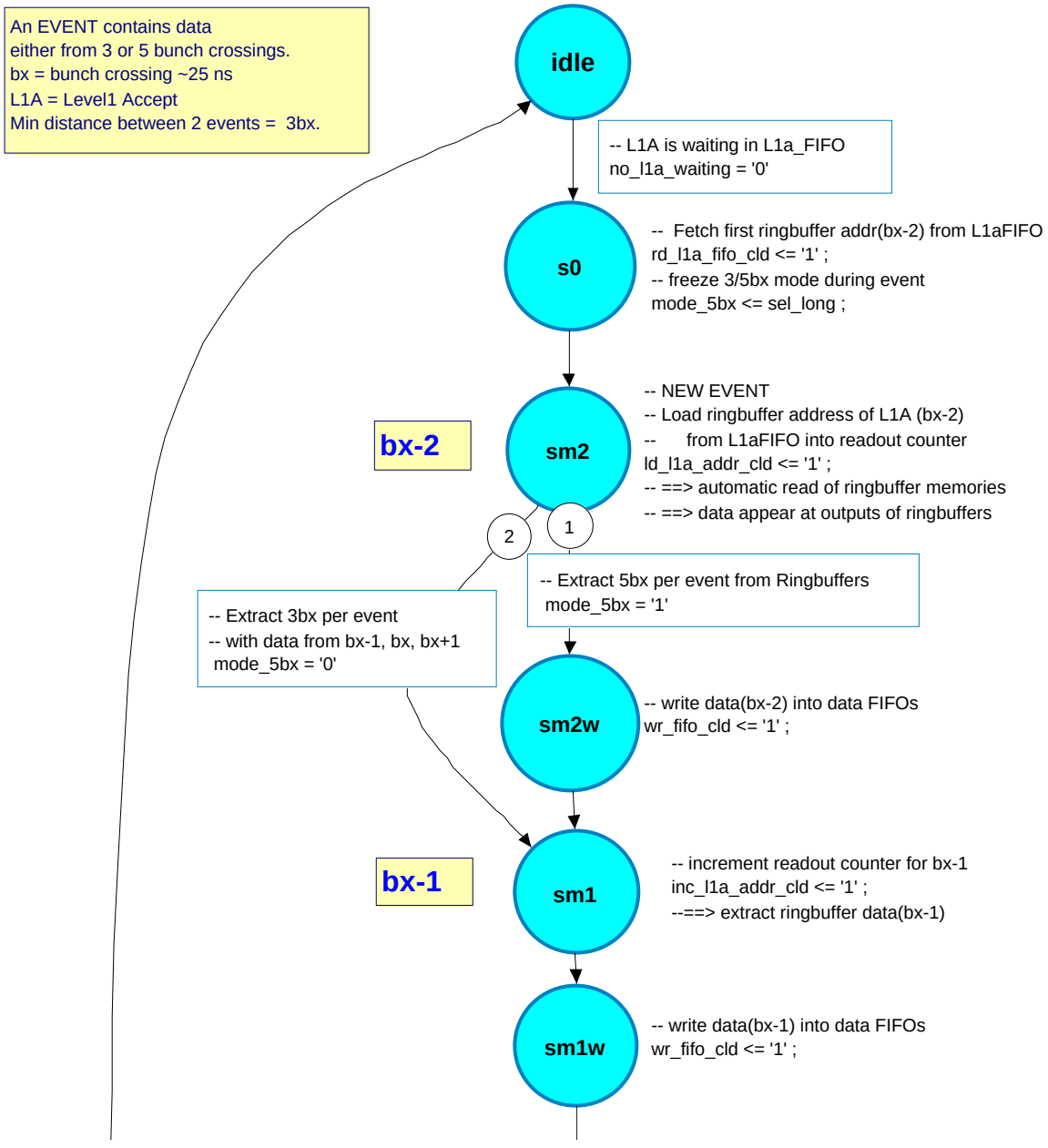
**Package List**
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

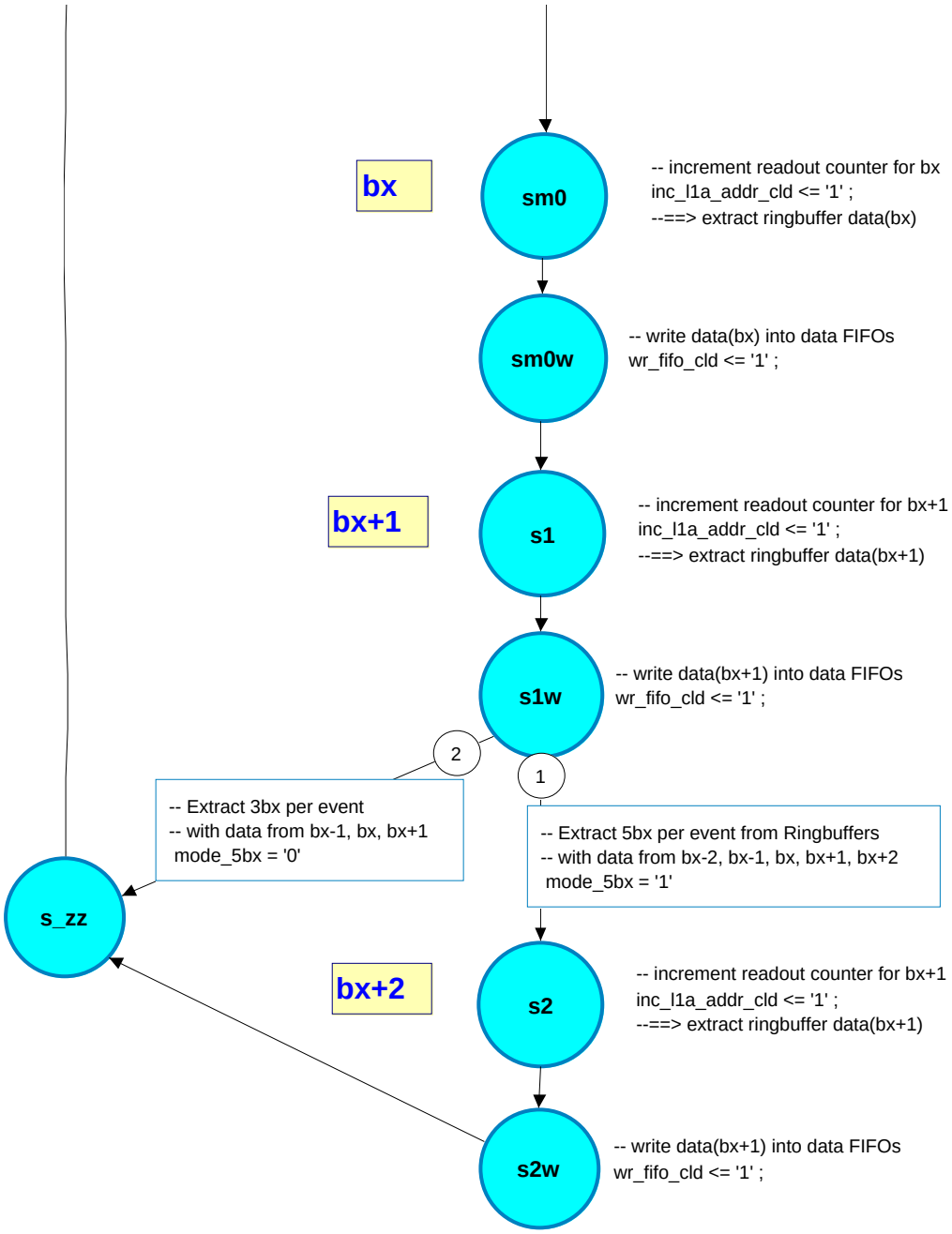
**Signal Status**

| SIGNAL | MODE | DEFAULT | RESET | SCHEME |
|---|---|---|---|---|
| inc_l1a_addr | OUT | '0' | '0' | CLKD |
| ld_l1a_addr | OUT | '0' | '0' | CLKD |
| wr_fifo | OUT | '0' | '0' | CLKD |
| rd_l1a_fifo | OUT | '0' | '0' | CLKD |
| mode_5bx | LOCAL | | '0' | CLKD |
| event_5bx | OUT | | | COMB |




| <company name> | Project: fdl_chip |
|---|---|
| Title: <enter diagram title here> | <enter comments here> |
| Path: v6_fdl_chip/l1a_sm/fsm | |
| Edited: by taurok on 08 Sep 2009 | |






**L1A_STATE MACHINE controls data extraction**
**from Ringbuffers for 3 & 5 bx events and writes them into data FIFOs.**
**If L1A is waiting then**
**--> ld l1a addr into the readout counter(var_length cntr)**
**--> read ringbuffer & write into data fifos**
**-- incr. var_lenth cntr**
**--> read ringbuffer & write into data fifos**
**etc until all bx data are moved into the FIFOS.**
**For 3bx-event mode do not extract bx-2 and bx+2 data.**

**Declarations**

**Ports:**

```
algo_s                : std_logic_vector(5 DOWNTO 0)
apply_new_presc_value : std_logic
clear                 : std_logic
clk                   : std_logic
en_algo_s             : std_logic_vector(5 DOWNTO 0)
en_rate_cntrs         : std_logic
finor_mask            : std_logic_vector(7 DOWNTO 0)
refresh               : std_logic_vector(7 DOWNTO 0)
update_period         : std_logic_vector(7 downto 0)
vme_data_in           : std_logic_vector(31 downto 0)
vme_en_noalgo_presc   : std_logic
rate_count            : std_logic_vector(31 DOWNTO 0)
to_ringbuff           : std_logic
to_sim_spy            : std_logic
trigger_bit           : std_logic_vector(7 DOWNTO 0)
```

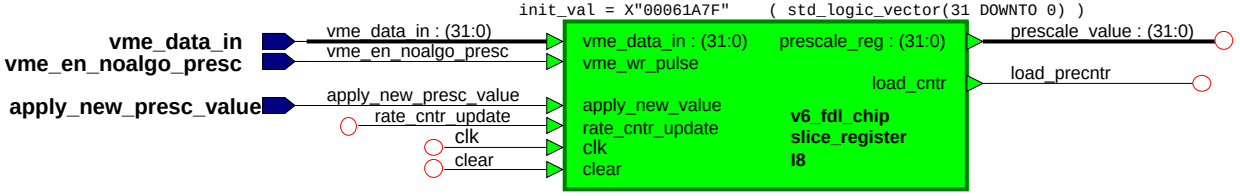
**Diagram Signals:**

```
SIGNAL bit_in               : std_logic
SIGNAL load_precntr         : std_logic
SIGNAL prescale_value       : std_logic_vector(31 DOWNTO 0)
SIGNAL prescaled_trigger    : std_logic
SIGNAL rate_cntr_update     : std_logic
SIGNAL rst_rate             : std_logic
```

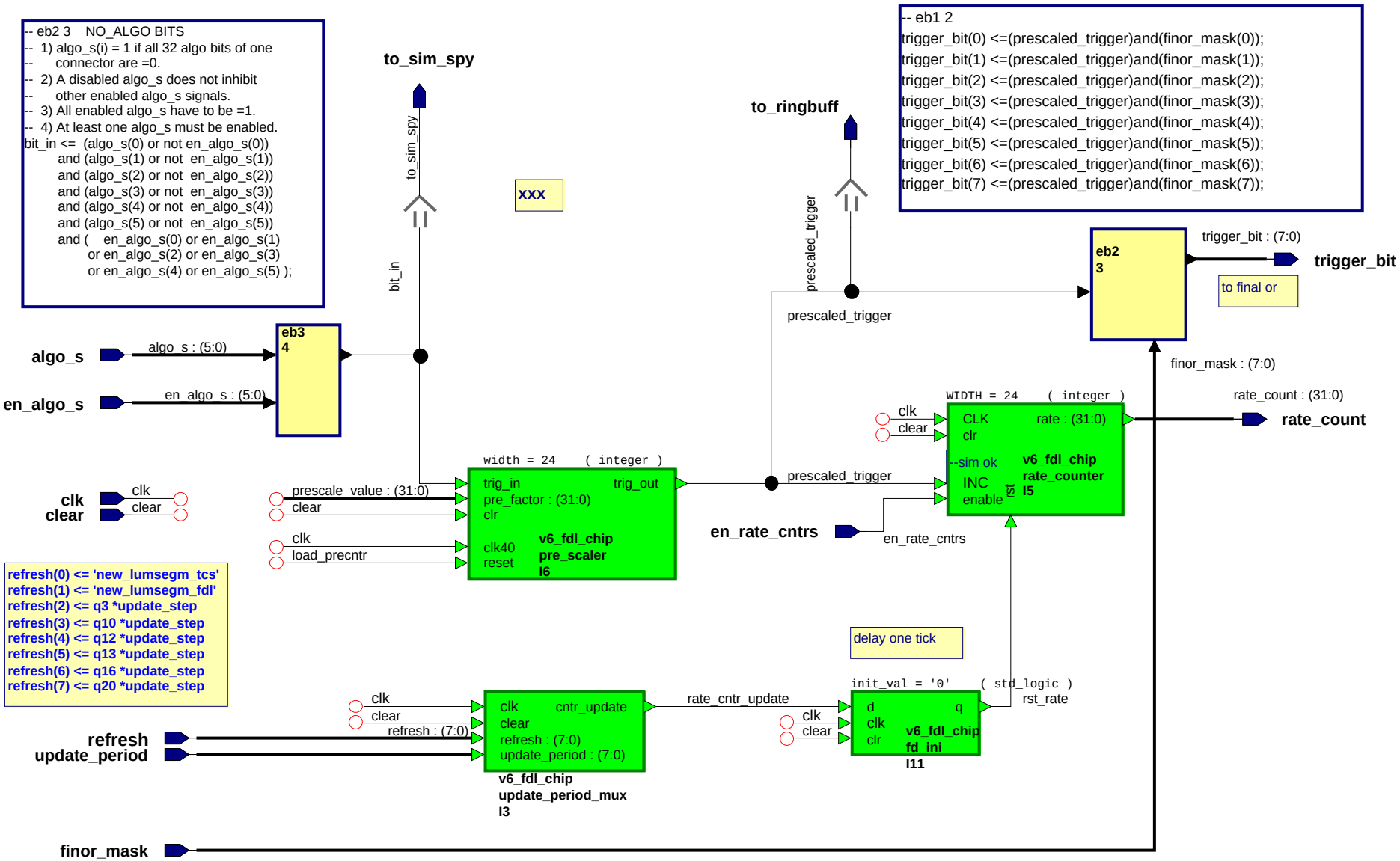
**Package List**

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
USE ieee.numeric_std.all;
```

Downscaling from 40 MHz to 100 Hz:
40MHz/100 = 400.000 = X"00061A7F"




| | | | |
|---|---|---|---|
| <company name> | | Project: | fdl_chip |
| | | 'enable' signal added to rate counter | |
| Title: | <enter diagram title here> | | |
| Path: | v6_fdl_chip/noalgo_slice/struct | | |
| Edited: | by taurok on 02 Sep 2009 | | |


-- eb2 3 NO_ALGO BITS
-- 1) algo_s(i) = 1 if all 32 algo bits of one
-- connector are =0.
-- 2) A disabled algo_s does not inhibit
-- other enabled algo_s signals.
-- 3) All enabled algo_s have to be =1.
-- 4) At least one algo_s must be enabled.
bit_in <= (algo_s(0) or not en_algo_s(0))
and (algo_s(1) or not en_algo_s(1))
and (algo_s(2) or not en_algo_s(2))
and (algo_s(3) or not en_algo_s(3))
and (algo_s(4) or not en_algo_s(4))
and (algo_s(5) or not en_algo_s(5))
and ( en_algo_s(0) or en_algo_s(1)
or en_algo_s(2) or en_algo_s(3)
or en_algo_s(4) or en_algo_s(5) );
to_sim_spy
to_sim_spy
bit_in
xxx
to_ringbuff
prescaled_trigger
prescaled_trigger
-- eb1 2
trigger_bit(0) <=(prescaled_trigger)and(finor_mask(0));
trigger_bit(1) <=(prescaled_trigger)and(finor_mask(1));
trigger_bit(2) <=(prescaled_trigger)and(finor_mask(2));
trigger_bit(3) <=(prescaled_trigger)and(finor_mask(3));
trigger_bit(4) <=(prescaled_trigger)and(finor_mask(4));
trigger_bit(5) <=(prescaled_trigger)and(finor_mask(5));
trigger_bit(6) <=(prescaled_trigger)and(finor_mask(6));
trigger_bit(7) <=(prescaled_trigger)and(finor_mask(7));
eb2
3
trigger_bit : (7:0)
trigger_bit
to final or
finor_mask : (7:0)
algo_s
algo_s : (5:0)
en_algo_s
en_algo_s : (5:0)
eb3
4
WIDTH = 24 ( integer )
clk
clear
CLK rate : (31:0)
clr
--sim ok
INC
enable
rst
v6_fdl_chip
rate_counter
I5
rate_count : (31:0)
rate_count
clk
clear
width = 24 ( integer )
prescale_value : (31:0)
clear
clk
load_precntr
trig_in trig_out
pre_factor : (31:0)
clr
clk40
reset
v6_fdl_chip
pre_scaler
I6
prescaled_trigger
en_rate_cntrs
en_rate_cntrs
refresh(0) <= 'new_lumsegm_tcs'
refresh(1) <= 'new_lumsegm_fdl'
refresh(2) <= q3 *update_step
refresh(3) <= q10 *update_step
refresh(4) <= q12 *update_step
refresh(5) <= q13 *update_step
refresh(6) <= q16 *update_step
refresh(7) <= q20 *update_step
delay one tick
clk
clear
refresh : (7:0)
clk cntr_update
clear
refresh : (7:0)
update_period : (7:0)
v6_fdl_chip
update_period_mux
I3
rate_cntr_update
clk
clear
init_val = '0' ( std_logic )
d q
clk
clr
v6_fdl_chip
fd_ini
I11
rst_rate
refresh
update_period
finor_mask

**Package List**

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
USE ieee.numeric_std.all;
USE IEEE.VITAL_Timing.all;

**Declarations**

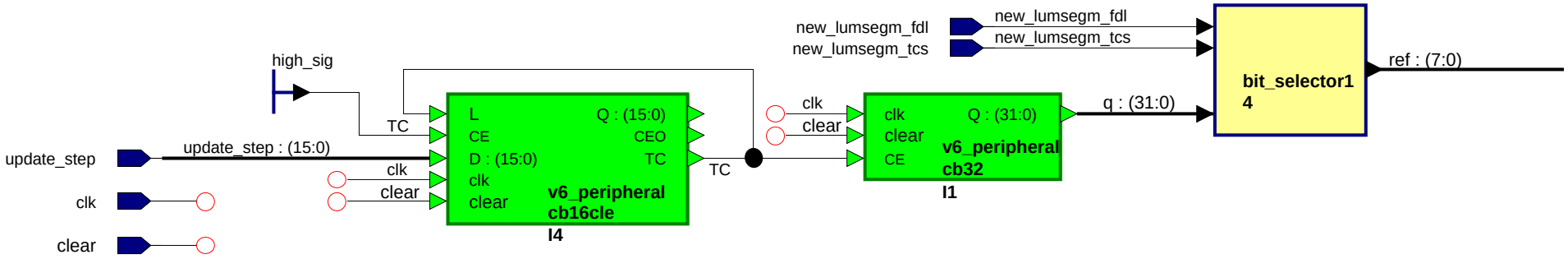
**Ports:**

```
clear           : std_logic
clk             : std_logic
new_lumsegm_fdl : std_logic
new_lumsegm_tcs : std_logic
update_step     : std_logic_vector(15 DOWNTO 0)
refresh         : std_logic_vector(7 DOWNTO 0)
```

**Diagram Signals:**

```
SIGNAL TC               : std_logic
SIGNAL high_sig         : std_logic
SIGNAL q                : std_logic_vector(31 DOWNTO 0)
SIGNAL q1               : std_logic_vector(7 DOWNTO 0)
SIGNAL ref              : std_logic_vector(7 DOWNTO 0)
```

```
-- eb3 3
ref(0)<=new_lumsegm_tcs;
ref(1)<=new_lumsegm_fdl;
ref(2)<=Q(3);
ref(3)<=Q(10);
ref(4)<=Q(12);
ref(5)<=Q(13);
ref(6)<=Q(16);
ref(7)<=Q(20);
```

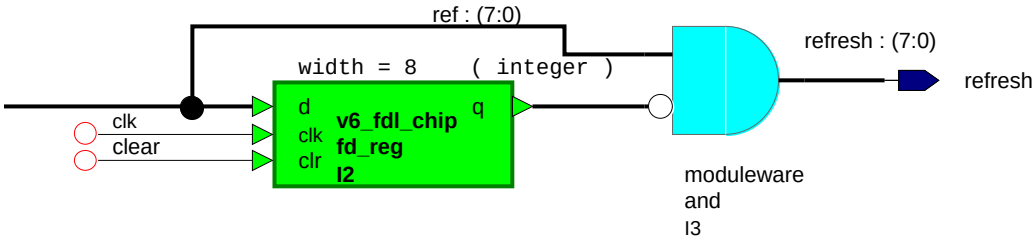



16 bit counter for generating a periodic TC signal (100 µs). update_step comes from a vme register. Load 4000 (=0xFA0) for 100 µs.

32 bit binary counter. At its output there are 32 periodic signals which can act as source for the final rate counter refresh signal.

bit_selector is used for selecting 7 out of 32 possible sources. One source is the refresh_from_tcs signal which comes over front panel.

| <company name> | | Project: | fdl_chip |
|---|---|---|---|
| Title: | <enter diagram title here> | <enter comments here> | |
| Path: | v6_fdl_chip/rate_counter_refresh/struct | | |
| Edited: | by taurok on 09 Sep 2009 | | |




This circuit generates a pulse from the periodic counter signals.

```
--  Copied from CMS Note 02_033: page 20
--    T3      T2   T1        T0
--     (Ready)(Busy)(OutofSync)(Warning)
--     PARTITION STATUS
--     0 0 0 0 DISCONNECTED *
--     0 0 0 1 OVERFLOW WARNING
--     0 0 1 0 OUT_OF_SYNC
--     0 0 1 1 forbidden
--     0 1 0 0 BUSY
--     0 1 0 1 forbidden
--     0 1 1 0 forbidden
--     0 1 1 1 forbidden
--     1 0 0 0 READY
--     1 0 0 1 forbidden
--     1 0 1 0 forbidden
--     1 0 1 1 forbidden
--     1 1 0 0 ERROR
--     1 1 0 1 forbidden
--     1 1 1 0 forbidden
--     1 1 1 1 DISCONNECTED *
```

**Package List**

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
USE IEEE.VITAL_Timing.all;
```

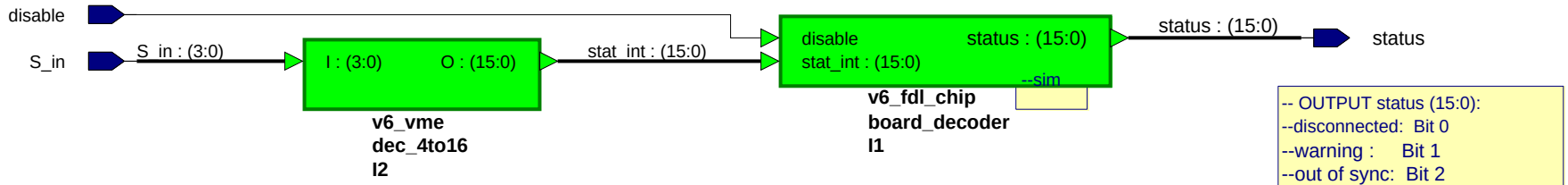
**Declarations**

**Ports:**

```
    S_in     : std_logic_VECTOR(3 DOWNTO 0)
    disable  : std_logic
    status   : std_logic_vector(15 DOWNTO 0)
```

**Diagram Signals:**

```
    SIGNAL stat_int : std_logic_VECTOR(15 DOWNTO 0)
```




```
bad_code <=    stat_int(3) or stat_int(5) or stat_int(6)  or stat_int(7)  or stat_int(9)  or stat_int(10) or stat_int(11) or stat_int(13) or stat_int(14);
```

```
if disable='0' then
   status(0) <= stat_int(0) or stat_int(15); -- disconnected
   status(1) <= stat_int(1);   -- warning
   status(2) <= stat_int(2);   -- out_of_sync
   status(3) <= stat_int(4);   -- busy
   status(4) <= stat_int(8);   -- ready
   status(5) <= stat_int(12);  -- error
  status(6)<='0';
  status(7)<= bad_code;
  status(15 downto 8) <= "00000000";
else
  status <= X"0010"; -- set status to ready if board is disabled
end if;
```

```
-- OUTPUT status (15:0):
--disconnected:  Bit 0
--warning :      Bit 1
--out of sync:  Bit 2
--busy:           Bit 3
--ready:          Bit 4
--error:          Bit 5
-- not used     Bit 6    : =0
--bad code:     Bit 7
-- not used     Bits 15-8: =0
```

| | | | |
|---|---|---|---|
| <company name> | | Project: | fdl_chip |
| | | <enter comments here> | |
| Title: | <enter diagram title here> | | |
| Path: | v6_fdl_chip/fastsig2stat/struct | | |
| Edited: | by taurok on 09 Sep 2009 | | |